AMIGA® OS 3.1

DOS



Commodore





COPYRIGHT

HAFTUNGSAUSSCHLUSS

WARENZEICHEN

Coverdesign und Druck im Selbstverlag von Village Tronic
Village Tronic Marketing GmbH, Wellweg 95, 31157 Sarstedt, Germany

Das vorliegende Buch wurde von Kitsel Outlaw, Ross Hippely, Barbara Siwirski und Carina Ahren erstellt unter Benutzung verschiedener Commodore-Rechnersysteme.

Inhaltsverzeichnis

Kapitel 1
Auswählen einer Benutzeroberfläche

Kapitel 2
AmigaDOS-Shell

Kapitel 3
Arbeiten mit AmigaDOS

Kapitel 4
Editoren

Kapitel 5
Befehlsdateien/Skripts

Kapitel 6
AmigaDOS-Befehlsreferenz

Kapitel 7
Referenz für Workbench-bezogene Befehle

Kapitel 8
Befehlsbeispiele

Anhang A
Fehlermeldungen

Anhang B
Zusätzliche Amiga-Verzeichnisse

Anhang C
Reine Diskettensysteme

Anhang D
Komplexere AmigaDOS-Funktionen

Glossar

Index



Willkommen!

Die Personal Computer der Commodore Amiga-Produktreihe bieten eine unübertroffene Kombination aus Vielseitigkeit, Rechenleistung und Benutzerfreundlichkeit. Das schnelle mit Multitasking ausgestattete Amiga-Betriebssystem ermöglicht allen Benutzern unabhängig von ihrem Kenntnisstand eine optimale Nutzung der Systemressourcen.

Bei AmigaDOS handelt es sich um das Disk-Betriebssystem des Amiga. Im Disk-Betriebssystem ist die Software zusammengefaßt, die zur Datenverarbeitung und zur Steuerung des Computers dient. Dazu zählen folgende Funktionen:

- Bereitstellung eines Dateisystems zur Verwaltung der Daten, die von Programmen verwendet und erzeugt werden
- Dateienverwaltung (Speichern und Abrufen) auf Disketten, Festplatten und anderen Speichermedien
- Organisation von Schnittstellen zu Peripheriegeräten (z. B. Drucker und Modems)

AmigaDOS stellt eine CLI-Benutzeroberfläche (Command Line Interface - Befehlszeileneingabe) zur Verfügung, so daß Sie über die direkte Eingabe von Befehlen mit dem Betriebssystem arbeiten können. Einige dieser Befehle entsprechen Ihnen bereits vertrauten Workbench-Operationen, wie z. B. COPY (Kopieren), RENAME (Umbenennen) und FORMAT (Disk formatieren). Erfahrenere Benutzer können außerdem mit erweiterten Befehlen sog. "Skripts" (Befehlsdateien) erstellen, die ihnen immer wiederkehrende Arbeitsschritte abnehmen. Darüber hinaus können Sie erweiterte Befehle verwenden, die zur Speicherverwaltung und zur Ausführung von Funktionen dienen, die nicht über die Workbench verfügbar sind.

Die Befehle werden in einem speziellen Fenster, dem sog. "Shell-Fenster", eingegeben. Shell-Fenster werden auf dem Workbench-Bildschirm geöffnet und sind mit Workbench-Fenstern vergleichbar. In Shell-Fenstern wird jedoch ausschließlich Text angezeigt.

Durch die Kombination von AmigaDOS und Amiga-Shell steht Ihnen eine leistungsfähige flexible Arbeitsumgebung zur Verfügung, die mit den in den folgenden Abschnitten beschriebenen Funktionen ausgestattet ist.

Betriebssystemfunktionen

- Vollständige Steuerung aller Bereiche des Amiga-Betriebs
- Hierarchisch aufgebautes Dateisystem
- Dateinamen bis zu 30 Zeichen lang; Groß- und Kleinschreibung wird zwar beibehalten, aber nicht berücksichtigt
- Konfigurierbare Befehlssuchpfade
- Namensmuster
- Befehlsverarbeitung im Hintergrund
- Viele interne Befehle und Möglichkeit, weitere Befehle speicher-resident zu machen
- Gemeinsam benutzte Funktionsbibliotheken (Shared Libraries)
- Unterstützung mehrerer Dateisysteme einschließlich CrossDOS (MS-DOS-Dateisystem)

Shell-Funktionen

- Mehrere voneinander unabhängige Shell-Fenster
- Shell-Fenster sind in der Größe einstellbar sowie verschiebbar, und sie können vom Hintergrund in den Vordergrund (bzw. umgekehrt) gebracht werden
- Eingabeaufforderung, Zeichensatz, Textfarbe und Schriftattribute sind konfigurierbar

- Befehlsvorgeschichte und Edieren von Befehlszeilen
- Schnelle zeichenorientierte Anzeige
- Aliase
- Lokale und globale Umgebungsvariablen
- Befehlsdateien (Skripts)
- Ein- und Ausgabeumleitung für Befehle
- Zuordnung mehrerer Verzeichnisse zu einem logischen
- Kopieren und Einfügen von Text von Konsolenfenster zu Konsolenfenster
- ARexx-Unterstützung

Verwendung dieses Handbuchs

In diesem Handbuch, das in Verbindung mit dem *Workbench-Benutzerhandbuch* zu verwenden ist, werden die AmigaDOS-Software, die zugehörigen Komponenten und deren Verwendung beschrieben. Bei den Erläuterungen wird davon ausgegangen, daß Sie sich zwar mit der Workbench auskennen, aber nie zuvor mit AmigaDOS gearbeitet haben. Ist dies der Fall, empfehlen wir Ihnen die Lektüre des kompletten Handbuchs, damit Sie sich vor den ersten Arbeitsschritten mit den Konzepten des Amiga-Betriebssystems vertraut machen. Später können Sie dieses Handbuch beim Ausführen von Befehlen oder beim Schreiben von Programmen oder Befehlsdateien (Skripts) als Referenz verwenden.

Das Handbuch besteht aus folgenden Kapiteln und Anhängen:

Kapitel 1. Auswählen einer Benutzeroberfläche: Anhand der in diesem Kapitel beschriebenen Kriterien können Sie leichter entscheiden, wann AmigaDOS anstelle der Workbench zu verwenden ist.

Kapitel 2. Merkmale der AmigaDOS-Shell: In diesem Kapitel wird die AmigaDOS-Shell detailliert beschrieben.

Kapitel 3. Arbeiten mit AmigaDOS: In diesem Kapitel werden das Dateiverwaltungssystem, die Befehlsarten sowie die Komponenten der AmigaDOS-Befehle erläutert.

Kapitel 4. Editoren: Dieses Kapitel enthält eine vollständige Erläuterung des Texteditors ED sowie Befehlslisten für die Texteditoren MEmacs und EDIT.

Kapitel 5. Befehlsdateien, Skripts: In diesem Kapitel werden die AmigaDOS-Skripts und deren Erstellung beschrieben.

Kapitel 6. AmigaDOS-Befehlsreferenz: In diesem Kapitel werden die einzelnen AmigaDOS-Befehle detailliert beschrieben.

Kapitel 7. Referenz für Workbench-bezogene Befehle: Dieses Kapitel enthält eine Beschreibung aller AmigaDOS-Befehle, die auch über die Workbench aufgerufen werden können.

Kapitel 8. Befehlsbeispiele: An den in diesem Kapitel enthaltenen Beispielen wird die Ausführung grundlegender Funktionen mit Hilfe von AmigaDOS-Befehlen beschrieben.

Anhang A. Fehlermeldungen: Dieses Kapitel enthält eine Liste der Probleme, die möglicherweise auftreten, und Vorschläge zu deren Behebung.

Anhang B. Zusätzliche Amiga-Verzeichnisse: In diesem Kapitel werden die Verzeichnisse S:, DEVS:, L:, FONTS: und weitere Verzeichnisse beschrieben.

Anhang C. Reine Diskettensystemem: In diesem Kapitel wird die optimale Nutzung von Systemen erläutert, die nur mit einem Diskettenlaufwerk und keiner Festplatte ausgestattet sind.

Anhang D. Spezielle AmigaDOS-Funktionen: Dieses Kapitel enthält Informationen zur benutzerspezifischen Anpassung von AmigaDOS für erfahrene Amiga-Benutzer.

Hinter Anhang D folgen ein Glossar und der Index.

Konventionen in diesem Handbuch

Im vorliegenden Handbuch gelten folgende Konventionen:

BEFEHLE, ZUGEORDNETE SCHLÜSSELWÖR-TER, GERÄTE und NAMEN	Befehle sowie deren Schlüsselwörter, Gerätenamen und die zugeordneten Verzeichnisse werden stets in Großbuchstaben geschrieben. Datei- und Verzeichnisnamen beginnen stets mit einem Großbuchstaben. Sie müssen jedoch nicht in dieser Form eingegeben werden. Der Amiga unterscheidet bei der Eingabe von Befehlen und Argumenten nicht zwischen Groß- und Kleinschreibung.
<n>	Spitze Klammern enthalten variable Daten, die vom Benutzer eingegeben werden müssen. Anstelle von <n> ist der/die jeweils gewünschte Wert, Text bzw. Option einzugeben. Die spitzen Klammern selbst werden nicht eingegeben.
`Courier`	Text im Zeichensatz Courier stellt Daten dar, die vom Benutzer in einem Fenster eingegeben oder als Antwort auf einen Befehl in einem Fenster angezeigt werden.
Taste1 - Taste2	Tastenkombinationen mit Bindestrich (-) bedeuten, daß die Tasten gleichzeitig zu drücken sind. Beispiel: Ctrl-C bedeutet, daß Sie die Taste Ctrl gedrückt halten und die Taste C drücken müssen.
Taste1, Taste2	Tastenkombinationen mit Komma (,) als Trennzeichen bedeuten, daß die Tasten nacheinander zu drücken sind. Beispiel: Esc,O bedeutet, daß Sie zunächst die Taste Esc drücken und loslassen und anschließend die Taste O.
Eingabetaste	Anweisungen zum Drücken der Eingabetaste bedeuten, daß Sie die große, außergewöhnlich geformte Taste oberhalb der rechten Umschalttaste drücken müssen.
Eingeben	Anweisungen zum "Eingeben" von Daten bedeuten, daß Sie die genannten Informationen eintippen und anschließend die Eingabetaste drücken.

Einrückung der Befehlszeilen Bei Befehlszeilen, die aufgrund ihrer Länge einen Zeilenumbruch enthalten, werden die folgenden Zeilen nur zu Dokumentationszwecken eingerückt angezeigt. In der Praxis beginnen die umgebrochenen Zeilen in derselben Spalte wie die Shell-Eingabeaufforderung.

Weitere Dokumentationen

AmigaDOS Kurzreferenz

Workbench-Benutzerhandbuch

ARexx-Benutzerhandbuch

Darüber hinaus enthalten die bei Addison-Wesley erschienenen *ROM-Kernel-Manuals* technische Dokumentationen zum Amiga-Betriebssystem für Programmierer und Entwickler auf Englisch.

Kapitel 1

Auswählen einer Benutzeroberfläche

Der Amiga ist mit einer grafischen Benutzeroberfläche (GUI - Graphical User Interface) namens Workbench ausgestattet, von der aus die meisten AmigaDOS-Operationen ausgeführt werden können, ohne Shell-Fenster zu öffnen. Es gibt jedoch einige gute Gründe, sich mit der AmigaDOS-Befehlszeileneingabe (CLI) vertraut zu machen. Die direkte Verwendung von AmigaDOS hat unter anderem folgende Vorteile:

- **Persönliche Vorliebe**

  Einige Benutzer arbeiten lieber mit Text und Tastatur als mit Piktogrammen und Maus. Der Grund dafür liegt dabei entweder in der persönlichen Vorliebe oder in der Kenntnis anderer textorientierter Computersysteme.

- **Grenzen der Workbench**

  Die meisten grundlegenden Operationen können zwar mit dergleichen Benutzerfreundlichkeit über die Shell oder die Workbench ausgeführt werden, aber für einige Funktionen sind AmigaDOS-Befehle erforderlich. Dazu gehören bestimmte grundlegende Funktionen zur Systemkonfiguration und zur Ausführung von Skripts und Hilfsprogrammen, für die keine Piktogramme vorhanden sind. Dabei ist zu beachten, daß über den Menüpunkt "Befehl ausführen" des Workbench-Menüs alle AmigaDOS-Funktionen auf der Workbench zur Verfügung stehen.

- **Geschwindigkeit**

  Benutzer, die über eine hohe Tippgeschwindigkeit verfügen und sich mit AmigaDOS auskennen, können gegebenenfalls einen

Befehl schneller über die Tastatur eingeben als die entsprechende Operation mit Hilfe der Maus aufzurufen. Dies gilt insbesondere, wenn mehrere Befehle ausgeführt werden sollen. Die Shell bietet Möglichkeiten wie z. B. Namensmuster und Ein-/Ausgabeumleitung, die im Vergleich zu Workbench-Methoden zu einer erheblichen Arbeitserleichterung führen. Darüber hinaus werden bei AmigaDOS-Befehlen die Textausgaben in der Regel schneller angezeigt als mit Piktogrammen gefüllte Dialogfenster und Fenster.

- **Steuerung und Flexibilität**

  Beim Aufrufen von Programmen von der Shell aus können die Multitasking-Funktionen des Amiga in der Regel leichter gesteuert werden. Dies gilt auch bei der Verwendung von Anwendungen wie Software-Compilern mit vielen Laufzeitoptionen. Häufig zu ändernde Optionen können Sie schneller über die Befehlszeile eingeben als durch Edieren der Merkmale eines Piktogramms.

- **Erstellen von Befehlsdateien**

  Die Ausführung komplexer, sich wiederholender Aufgaben ohne Benutzereingriff ist über die Workbench schwierig, falls nicht sogar unmöglich. Für derartige Aufgaben eignen sich insbesondere Skripts, d. h. Textdateien mit AmigaDOS-Befehlen.

- **Einsparen von Ressourcen für Ihre Anwendungen**

  Für eine Textschnittstelle sind weniger Hauptspeicher, weniger Disk-Kapazität und weniger andere Systemressourcen erforderlich als für die Grafikkomponenten einer GUI.

1.1 Auswählen Ihrer Oberfläche

Obwohl einige Benutzer die ausschließliche Verwendung der Shell oder der Workbench bevorzugen, können die meisten beides verwenden, nachdem sie sich mit den Grundlagen von AmigaDOS vertraut gemacht haben. Da Shell-Fenster auf dem Workbench-Bildschirm geöffnet werden können, ist ein problemloser Wechsel zwischen den beiden Arbeitsmethoden möglich. Ob Sie die Workbench oder die Shell verwenden, hängt dann vom Einzelfall und Ihrer Einschätzung ab, mit welcher Methode Sie bestimmte Funktionen schneller ausführen können.

1.1.1 Workbench-Benutzer

Sie können zwar primär mit der Workbench arbeiten, aber wir empfehlen Ihnen dennoch, sich mit AmigaDOS vertraut zu machen, damit Sie spezielle AmigaDOS-Befehle aufrufen und gegebenenfalls Befehlsdateien auf deren Funktion überprüfen können. Die zahlreichen benutzerfreundlichen Funktionen der Amiga-Shell vereinfachen den Lernprozeß und ermöglichen im Vergleich zu anderen Befehlszeilensystemen eine wesentlich leichtere Verwendung von AmigaDOS. Falls Sie es bevorzugen, AmigaDOS nicht direkt zu verwenden, können Sie Befehlsdateien und Shell-Befehlen Piktogramme zuordnen.

1.1.2 Shell-Benutzer

Shell-Benutzer, die nicht mit der Workbench arbeiten, müssen dennoch nicht auf die Vorteile der integrierten GUI-Unterstützung des Amiga verzichten. Mit Hilfe der Maus können Shell-Fenster - wie Workbench-Fenster - schnell verschoben und vergrößert/verkleinert werden. Außerdem können Shell-Fenster jederzeit in den Vorder- bzw. Hintergrund gebracht und jederzeit geöffnet oder geschlossen werden. Shell-Fenster können auch auf anderen "öffentlichen" Bildschirmen als dem Workbench-Bildschirm geöffnet werden. Darüber hinaus können Sie die Arbeitsumgebung für die Befehlszeile unter Verwendung der Hilfsprogramme "FKey", "MouseBlanker" und anderer Commodity-Programme Ihren Wünschen entsprechend anpassen.

1.2 AmigaDOS-Funktionen

Bei der Entscheidung, mit welcher Amiga-Benutzeroberfläche Sie arbeiten, können Sie der folgenden Tabelle die Kapitel entnehmen, in denen die einzelnen Funktionen beschrieben werden. Vergleichen Sie anschließend die AmigaDOS-Methode mit der Workbench-Methode. Wählen Sie die Methode aus, mit der Sie Ihr Ziel am einfachsten und benutzerfreundlichsten erreichen.

Konfigurieren über Voreinsteller	Erläuterung in
Auswählen einer Sprache	Kapitel 7
Auswählen eines Tastaturtyps und der Mausoptionen	Kapitel 7
Auswählen des Standardbildschirmmodus	Kapitel 7
Edieren der Workbench-Farben	Kapitel 7
Einstellen Systemdatum und -uhrzeit	Kapitel 7
Auswählen der Systemzeichensätze	Kapitel 7
Auswählen der Druckeroptionen	Kapitel 7
Edieren des Mauszeigers	Kapitel 7
Auswählen der Workbench-Hintergrundmuster	Kapitel 7
Wahl des Tons für akustische Signale	Kapitel 7

Konfigurieren der Disks und der Arbeitsumgebung	Erläuterung in
Formatieren und Kopieren von Disks	Kapitel 7
Hinzufügen von Verzeichnissen zum Suchpfad	Kapitel 6, 8
Erstellen von Alias-Namen	Kapitel 6, 8, Anhang C
Zuordnen von Verzeichnissen und Geräten	Kapitel 6, 8, Anhang C

Konfigurieren der Disks und der Arbeitsumgebung (Forts.)	Erläuterung in (Forts.)
Verwenden der Option ASSIGN PATH	Kapitel 6, Anhang C
Befehle speicherresident machen	Kapitel 6, Anhang C
Anpassen der Startup-Dateien	Anhang D
Erstellen neuer Verzeichnisse	Kapitel 6
Einstellen von CrossDOS-Optionen	Kapitel 7
Konfigurieren der Funktionstasten	Kapitel 7
Vorbereiten von PCMCIA-Speicherkarten für deren Verwendung	Kapitel 6, 7
Vorheriges Laden von Ressourcen in den Speicher	Kapitel 6, Anhang C
Freimachen von Kapazität auf der Workbench-Disk	Anhang C
Verwenden von Mount-Dateien und MountLists	Kapitel 6, Anhang B

Erlernen von AmigaDOS und Shell	Erläuterung in
Entscheiden, wann AmigaDOS verwendet werden soll	Kapitel 1
Öffnen und Schließen von Shell-Fenstern	Kapitel 2, 6, 8
Aktivieren eines Shell-Fensters	Kapitel 2
Grundlagen von Befehlen	Kapitel 6
Befehlszeilenformat	Kapitel 6
Befehlsschablonen	Kapitel 6
AmigaDOS-Sonderzeichen	Kapitel 3
Namensmuster und Jokerzeichen	Kapitel 3

Erlernen von AmigaDOS und Shell (Forts.)	Erläuterung in (Forts.)
Edieren der Befehlszeile	Kapitel 2
Verwenden der Befehlsvorgeschichte	Kapitel 2
Aufrufen früherer Befehlsausgaben	Kapitel 2, 8
Verwenden von Kopieren und Einfügen	Kapitel 2
Angeben von Pfaden	Kapitel 3, 8
Arbeiten mit einer einzigen Shell	Kapitel 8
Auf Disk residierende und interne Befehle	Kapitel 3
Anpassen des Shell-Fensters	Kapitel 6, 8, Anhang D
Standardverzeichnisstruktur	Anhang B
Benennen und Umbenennen von Disks, Dateien und Verzeichnissen	Kapitel 3, 6

Anzeigen von Daten	Erläuterung in
Feststellen des aktuellen Verzeichnisses	Kapitel 3, 8
Auflisten des Verzeichnisinhalts (Beispiel)	Kapitel 6, 8
Anzeigen einer Befehlsschablone	Kapitel 3
Auflisten der Informationen zu Dateien und Verzeichnissen	Kapitel 6, 8
Verwenden von LIST LFORMAT (Beispiel)	Kapitel 6
Anzeigen oder Einstellen von Datum und Uhrzeit	Kapitel 7
Anzeigen von Grafiken, Text und Animationsdateien	Kapitel 6, 7
Verwenden eines Bildschirmtaschenrechners	Kapitel 7
Verwenden einer Bildschirmuhr	Kapitel 7
Anzeigen der Tastaturbelegung	Kapitel 7
Anzeigen der Software-Versionsnummern	Kapitel 6
Ändern des Zeichensatzes für Shell-Fenster	Kapitel 6

Anzeigen von Daten (Forts.)	Erläuterung in (Forts.)
Aufrufen der Informationen zu Dateisystemen	Kapitel 6
Verwenden von Escape-Sequenzen	Anhang D
Auflisten der Informationen zu Shell-Prozessen	Kapitel 6

Aufrufen von Programmen	Erläuterung in
Aufrufen von Programmen von der Shell aus	Kapitel 3, 8
Angeben des richtigen Pfads	Kapitel 3, 8
Wechseln des aktuellen Verzeichnisses	Kapitel 8
Umleiten der Befehlsein- und -ausgabe	Kapitel 3
Stoppen eines Programms (Beispiel)	Kapitel 8
Ausführen von Befehlen als Hintergrundprozesse	Kapitel 6, 8
Verwenden von ICONX (Beispiel)	Kapitel 8

Befehlsdateien, Skripts	Erläuterung in
Aufrufen von Befehlsdateien	Kapitel 5, 6
Verlassen einer Befehlsdatei	Kapitel 6
Edieren von Text und Befehlsdateien	Kapitel 4
Erstellen und Ändern einer benutzerspezifischen Startup-Datei (User-startup)	Kapitel 8, Anhang D
Verwenden von Skript-Sonderzeichen	Kapitel 5
Setzen des s-Schutzbits	Kapitel 5, 8
Automatisches Erstellen von Befehlsdateien	Kapitel 5, 6, 8
Verwenden von PCD	Kapitel 8, Anhang B
Verhindern der Bildschirmausgabe durch >NIL:	Kapitel 8
Skript-Befehle	Kapitel 5
Skripts zur Fehlersuche	Kapitel 5

Befehlsdateien, Skripts (Forts.)	Erläuterung in (Forts.)
Umgebungsvariablen	Kapitel 5
Erstellen von Schleifen mit Hilfe von EVAL (Beispiel)	Kapitel 8
Erstellen eines Befehls MOVE (Verlegen)	Kapitel 8

Texteditoren	Erläuterung in
ED (Editor)	Kapitel 4
Zugreifen auf erweiterte ED-Menüs	Kapitel 8
MEmacs (Editor)	Kapitel 4
EDIT (Editor)	Kapitel 4

Bearbeiten von Dateien	Erläuterung in
Kopieren von Dateien oder Verzeichnissen	Kapitel 6, 8
Kopieren von Disks	Kapitel 7
Löschen von Dateien oder Verzeichnissen	Kapitel 6, 8
Löschen von Dateien mit Piktogrammen	Kapitel 8
Löschen mit interaktivem Befehl DIR	Kapitel 8
Sortieren und Verknüpfen von Dateien	Kapitel 8
Testen von Befehlen	Kapitel 8
RAM-Disk	Anhang C
Entfernen nicht verwendeter Zeichensätze und Bibliotheken aus dem Speicher	Kapitel 8
Pipes (Beispiel)	Kapitel 8, Anhang B
Berechnen einfacher Ausdrücke	Kapitel 6
Suchen bestimmter Zeichenfolgen	Kapitel 6

Bearbeiten von Dateien (Forts.)	**Erläuterung in (Forts.)**
Alphabetisches Sortieren der Zeilen in einer Datei	Kapitel 6
Umleiten der Druckerausgabe	Kapitel 7
Aktualisieren von .font-Dateien	Kapitel 7

Workbench-spezifische Funktionen	**Kapitel**
Angeben von Workbench-Parametern	Kapitel 7
Edieren von Piktogrammen	Kapitel 7
Zuordnen von Piktogrammen zu Dateien	Kapitel 8
Erstellen eines benutzerspezifischen RAM-Disk-Piktogramms	Kapitel 8
Verwenden von ICONX	Kapitel 8
Verwenden eines Bildschirmtaschenrechners	Kapitel 7
Verwenden einer Bildschirmuhr	Kapitel 7
Ausdrucken eines Bildschirms	Kapitel 7
Erstellen von Workbench-Hintergrundmustern	Kapitel 7
Ausblenden der Bildschirmanzeige	Kapitel 7
Ausblenden des Mauszeigers	Kapitel 7
Ändern der Workbench-Farben	Kapitel 7
Ändern des Mauszeigers	Kapitel 7
Einstellen der Eingabeoptionen	Kapitel 7
Angeben von Zeichensätzen	Kapitel 7
Auswählen der Anzeigemodi	Kapitel 7
Starten der Workbench von der Shell aus	Kapitel 6



Kapitel 2
AmigaDOS-Shell

Bei der AmigaDOS-Shell handelt es sich um ein spezielles Fenster auf dem Workbench-Bildschirm, in das Sie Text zur Kommunikation mit AmigaDOS eingeben können. Die Shell ist eine Art Benutzeroberfläche mit Befehlszeileneingabe (CLI - Command Line Interface). In diesem Kapitel werden folgende Themen behandelt:

- Einführung in die Shell
- Öffnen und Schließen von Shell-Fenstern
- Verwenden der Shell

2.1 Einführung in die Shell

Über ein Shell-Konsolenfenster können Sie direkt mit AmigaDOS kommunizieren. Ein Konsolenfenster ist eine reine Textschnittstelle, von der Eingaben über die Tastatur akzeptiert werden. Mit folgenden Ausnahmen entsprechen Shell-Fenster in Aussehen und Funktionsweise Workbench-Fenstern:

- Piktogramme können nicht in Shell-Fenster gezogen werden.
- Die Maus kann außer bei den Texteditoren ED und MEmacs nur zum Kopieren und Einfügen verwendet werden.
- In Shell-Fenstern werden keine Rollsymbole angezeigt.
- Im AmigaDOS-Shell-Fenster kann nur ein Zeichensatz mit festem Zeichenabstand verwendet werden. Dabei handelt es sich in der Regel um den im Zeichensatz-Voreinsteller (Font) angegebenen Zeichensatz für Standard-System-Texte (Topaz oder Courier).

- Im Workbench-Muster-Voreinsteller (WBPattern) angegebene Workbench-Hintergrundmuster werden in Shell-Fenstern nicht angezeigt.

Abb. 2-1 zeigt ein Shell-Fenster, das auf dem Workbench-Bildschirm geöffnet wurde.

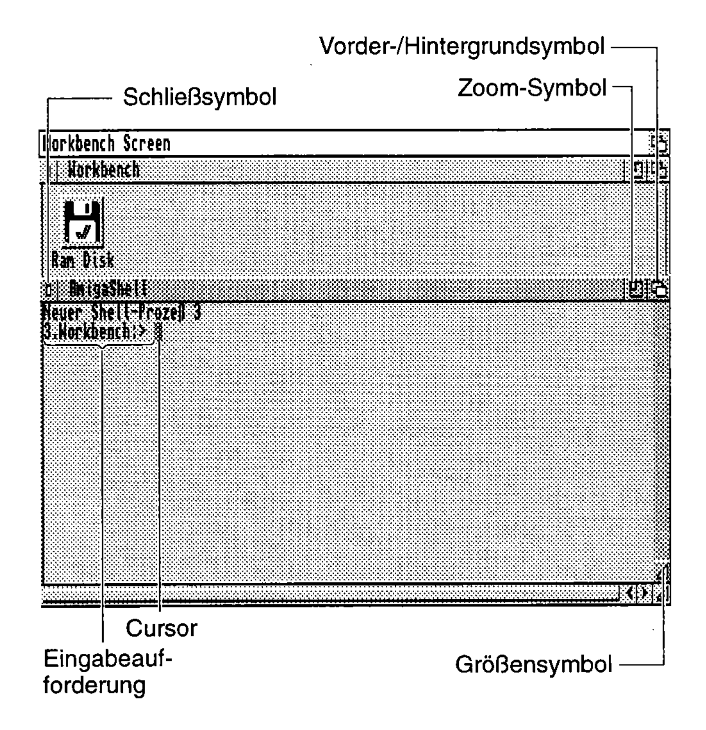


Abb. 2-1. Shell-Fenster

Wie bei der Workbench können gleichzeitig mehrere, voneinander unabhängige Shell-Fenster geöffnet sein. Während in einer Shell eingegebene Befehle ausgeführt werden, können Sie in einem anderen Shell-Fenster Befehle eingeben und vom Computer ausführen lassen.

2.2 Öffnen von Shell-Fenstern

Shell-Fenster können mit zwei Methoden geöffnet werden:

- Klicken Sie auf der Workbench das Piktogramm "Shell" in der Schublade "System" an.
- Geben Sie den in Kapitel 6 beschriebenen Befehl NEWSHELL ein.

Nach Öffnen eines Shell-Fensters geschieht folgendes:

- Das Fenster wird hervorgehoben um anzuzeigen, daß es sich um das aktuelle Fenster handelt.
- Eine Eingabeaufforderung wird angezeigt (z. B. `1.SYS:>`).
- Rechts hinter der Eingabeaufforderung erscheint ein kleines, hervorgehobenes Rechteck, der sog. "Cursor".

Wie bei der Workbench empfängt nur das aktuelle Fenster Eingabedaten. Sollen Daten in ein anderes Fenster eingegeben werden, klicken Sie das entsprechende Fenster an, um es zum aktuellen Fenster zu machen. Solange das aktuelle Fenster ein Shell-Fenster ist, stehen in der Workbench-Titelleiste keine Menüs zur Verfügung.

2.3 Schließen von Shell-Fenstern

Verwenden Sie zum Schließen eines Shell-Fensters eine der drei folgenden Methoden:

- Wählen Sie das Schließsymbol aus.
- Geben Sie den Befehl ENDSHELL ein.
- Drücken Sie die Tastenkombination Ctrl-\.

Nach Ausführung der gewünschten Befehle empfiehlt es sich, Shell-Fenster zu schließen. Ansonsten wird unnötig Speicherplatz belegt.

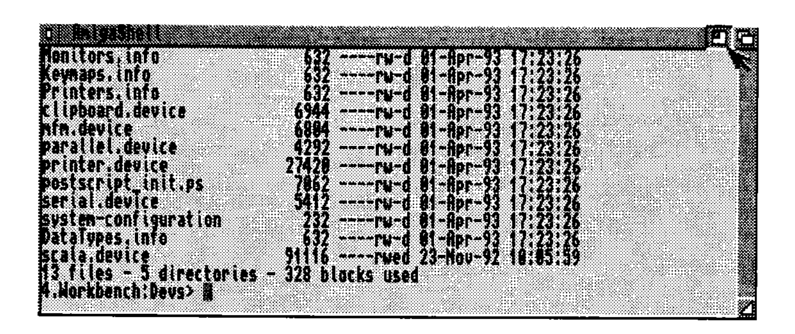
Alle nicht unabhängigen Programme, die von der Shell aus aufgerufen wurden, müssen vor Schließen eines Shell-Fensters geschlossen werden. Wird nach Drücken der Eingabetaste keine Shell-Eingabeaufforderung angezeigt, ist noch ein Programm aktiv. Sie können zwar weiterhin Befehle in ein derartiges Fenster eingeben,

aber AmigaDOS reagiert erst nach Verlassen des laufenden Programms wieder auf die eingegebenen Befehle.

2.4 Verwenden der Shell

Geben Sie die gewünschten AmigaDOS-Befehle hinter der Shell-Eingabeaufforderung ein. Bei jedem Befehl müssen Sie die gegebenenfalls zusätzlich erforderlichen Daten wie z. B. Dateinamen und Befehlsoptionen eingeben. Drücken Sie am Ende jeder Befehlszeile die Eingabetaste, damit der Befehl ausgeführt wird. Nach Ausführung des Befehls wird wieder die Shell-Eingabeaufforderung angezeigt.

Wenn die Ausgabedaten für einen Befehl aus dem Shell-Fenster rollen, vergrößern Sie das Fenster durch Auswahl des Zoom-Symbols der Shell oder unter Verwendung des Größensymbols. Daraufhin wird zusätzlich der Teil des vorherigen Fensterinhalts angezeigt, der jetzt in das Fenster paßt. Abb. 2-2 zeigt ein Shell-Fenster vor und nach Verwendung des Zoom-Symbols zur Anzeige der gesamten Ausgabedaten für den Befehl LIST (Auflisten).



Vorherige Ausgabedaten

```
AmigaShell
New Shell process 4
4.Workbench:> cd devs
4.Workbench:Devs> LIST
Monitors                 Dir ----rwed 19-Apr-93 15:55:05
DataTypes                Dir ----rwed 01-Apr-93 17:18:55
DOSDrivers               Dir ----rwed 19-Apr-93 15:55:05
Printers                 Dir ----rwed 20-Apr-93 15:55:05
Keymaps                  Dir ----rwed 19-Apr-93 16:02:43
DOSDrivers.info          632 ----rw-d 01-Apr-93 17:23:26
Monitors.info            632 ----rw-d 01-Apr-93 17:23:26
Keymaps.info             632 ----rw-d 01-Apr-93 17:23:26
Printers.info            632 ----rw-d 01-Apr-93 17:23:26
clipboard.device        6944 ----rw-d 01-Apr-93 17:23:26
mfm.device              6804 ----rw-d 01-Apr-93 17:23:26
parallel.device         4292 ----rw-d 01-Apr-93 17:23:26
printer.device         27420 ----rw-d 01-Apr-93 17:23:26
postscript_init.ps      7062 ----rw-d 01-Apr-93 17:23:26
serial.device           5412 ----rw-d 01-Apr-93 17:23:26
system-configuration     232 ----rw-d 01-Apr-93 17:23:26
DataTypes.info           632 ----rw-d 01-Apr-93 17:23:26
scala.device           91116 ----rwed 23-Nov-92 10:05:59
13 files - 5 directories - 328 blocks used
4.Workbench:Devs>
```

Abb. 2-2. Anzeigen vorheriger Ausgabedaten mit dem Zoom-Symbol

2.4.1 Edieren der Befehlszeile und Befehlszeilensteuerung

Zum leichteren Eingeben und Edieren von Befehlszeilentext stehen in der AmigaDOS-Shell folgende Textediertasten und Tastenkombinationen zur Verfügung:

Cursor links	Rückt den Cursor ein Zeichen nach links.
Cursor rechts	Rückt den Cursor ein Zeichen nach rechts.
Shift-Cursor links	Setzt den Cursor auf den Zeilenanfang.
Shift-Cursor rechts	Setzt den Cursor auf das Zeilenende.
Rücktaste	Löscht das Zeichen links des Cursors.
Del	Löscht das durch den Cursor hervorgehobene Zeichen.
Ctrl-H	Löscht das letzte Zeichen (Funktion entspricht Rücktaste).
Ctrl-M	Führt Befehlszeile aus (Funktion entspricht Eingabetaste).
Ctrl-J	Fügt Zeilenvorschub ein.
Ctrl-W	Löscht das Wort links des Cursors.
Ctrl-X	Löscht die aktuelle Zeile.
Ctrl-K	Löscht den gesamten Text von der Cursorposition bis zum Zeilenende.
Ctrl-Y	Ruft die mit Ctrl-K gelöschten Zeichen wieder auf.
Ctrl-U	Löscht den gesamten Text von der Cursorposition bis zum Zeilenanfang.

Darüber hinaus werden von der Shell folgende Tasten und Tastenkombinationen unterstützt:

Leertaste (oder jedes andere druckfähige Zeichen)	Ausgabe wird angehalten (Rollen wird unterbrochen).
Rücktaste	Ausgabe wird fortgesetzt (Rollen wird fortgesetzt).
Ctrl-C	Sendet einen BREAK-Befehl an den aktuellen Prozeß (Prozeß wird abgebrochen).

Ctrl-D	Sendet einen BREAK-Befehl an das aktuelle Skript (Skript wird abgebrochen).
Ctrl-F	Aktiviert Workbench-Programmfenster und bringt sie in den Vordergrund.
Ctrl-S	Ausgabe wird angehalten.
Ctrl-Q	Ausgabe wird fortgesetzt, wenn sie mit Ctrl-S angehalten wurde.
Ctrl-	Schließt das Shell-Fenster. Wenn die Konsolen-E/A (Ein-/Ausgabe) mit * an ein anderes Gerät umgeleitet wurde, wird die normale E/A wiederhergestellt.

In der Shell haben Sie die Möglichkeit, Befehle und andere Daten einzugeben, während Daten aufgelistet werden. Dadurch wird jedoch die Ausgabe unterbrochen, bis sie durch Drücken der Eingabetaste wieder fortgesetzt wird. Nach Beenden der Ausgabe wird der neue Befehl ausgeführt.

Wenn Sie einen neuen Befehl oder Text eingeben und sofort wieder löschen, wird die ursprüngliche Ausgabe fortgesetzt, sobald das letzte Zeichen gelöscht ist.

2.4.2 Verwenden der Befehlsvorgeschichte

Die Shell verfügt über einen Befehlszeilenpuffer mit einer Kapazität von 2 KB, der als Befehlsvorgeschichte dient. Mit Hilfe der Befehlsvorgeschichte können Sie zuvor eingegebene Befehlszeilen wieder aufrufen und edieren sowie die Befehle erneut starten. Auf diese Weise können Sie problemlos Befehle wiederholen oder mehrere ähnliche Befehle eingeben. Abb. 2-3 zeigt eine Reihe von Befehlen, die im Befehlszeilenpuffer gespeichert sind.

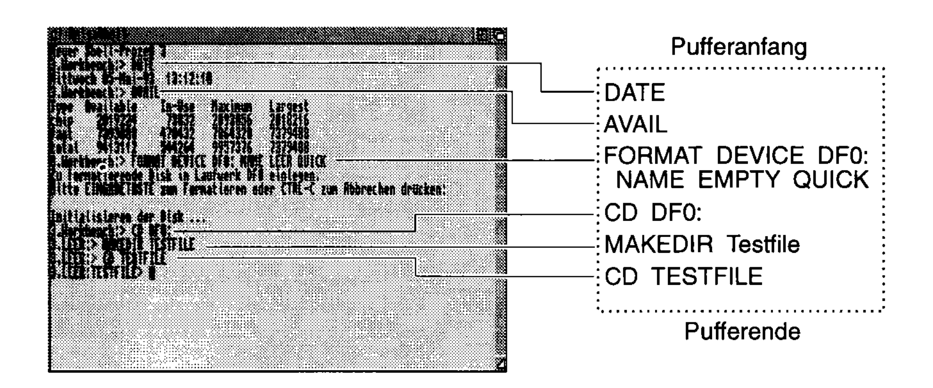


Abb. 2-3. Befehlszeilenpuffer

Die genaue Anzahl der im Befehlszeilenpuffer gespeicherten Zeilen hängt von der Länge der einzelnen Zeilen ab. Wenn der Puffer nicht mehr über genügend Speicherplatz verfügt, werden die zuerst gespeicherten Zeilen gelöscht. Mit den Tasten "Cursor aufwärts" und "Cursor abwärts" können Sie auf Befehlszeilen zugreifen:

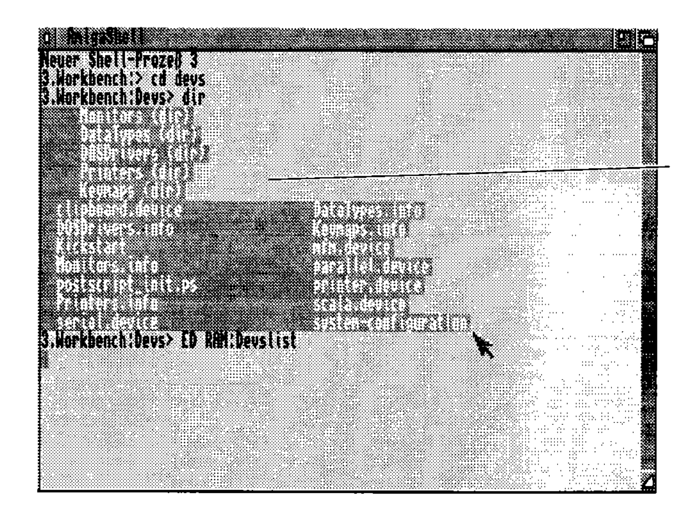
Cursor aufwärts	Früher im Befehlszeilenpuffer gespeicherte Zeilen abrufen.
Cursor abwärts	Später im Befehlszeilenpuffer gespeicherte Zeilen abrufen.

Wenn Sie z. B. mehrere .info-Dateien von einem Verzeichnis in ein anderes kopieren wollen, geben Sie die Zeile mit dem vollständigen Pfadnamen nur einmal ein. Anschließend rufen Sie diese Zeile so oft wie nötig auf und ändern in der Zeile jeweils nur den Namen der Datei.

Darüber hinaus können Sie in der Befehlsvorgeschichte nach dem letzten Vorkommen eines bestimmten Befehls suchen. Geben Sie dazu die zugehörige Befehlszeile bzw. deren Anfang ein, und drücken Sie anschließend die Tastenkombination Shift-"Cursor aufwärts" (oder Ctrl-R). Wenn Sie beispielsweise **DIR** eingeben und die Tastenkombination Shift-"Cursor aufwärts" drücken, wird die letzte Befehlszeile angezeigt, in der über den Befehl DIR der Inhalt eines Verzeichnisses abgefragt wurde. Mit der Tastenkombination Shift-"Cursor abwärts" gelangen Sie an das Ende des Befehlszeilenpuffers, wobei der Cursor in eine Leerzeile gesetzt wird.

2.4.3 Kopieren und Einfügen

Über die Shell können Sie auch Informationen aus einem Konsolenfenster (z. B. aus einem Shell- oder ED-Fenster) in ein anderes oder in dasselbe Fenster kopieren oder einfügen. Dies ist die einzige Workbench-ähnliche Mausoperation (abgesehen von den Texteditoren ED und MEmacs), die innerhalb von Shell-Fenstern ausgeführt werden kann. Abb. 2-4 zeigt das Kopieren und Einfügen von einem Shell-Fenster in ein ED-Fenster.



Hervorgehobener Text ("Rechte Amiga-Taste"+C zum Kopieren)

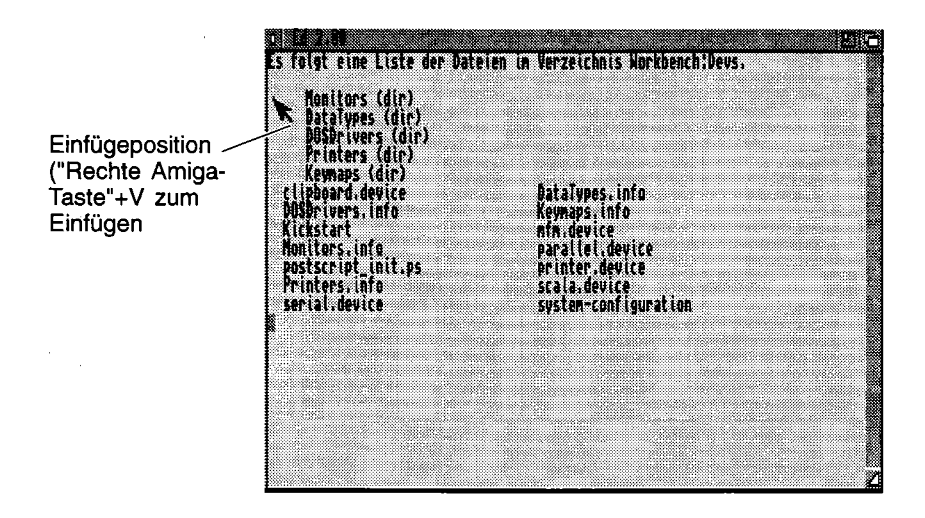


Abb. 2-4. Kopieren und Einfügen

Zum Kopieren und Einfügen von Daten markieren Sie den zu kopierenden und einzufügenden Text. Setzen Sie dazu den Mauszeiger an den Anfang des gewünschten Textbereichs, halten Sie die Auswahltaste der Maus gedrückt und ziehen Sie den Mauszeiger an das Ende des zu kopierenden Texts. Lassen Sie die Auswahltaste los und drücken Sie die Tastenkombination "Rechte Amiga-Taste"-C. Der markierte Text wird in die Zwischenablage kopiert und die Markierung rückgängig gemacht. Der kopierte Text kann wiederholt in Fenster von Anwendungen kopiert werden, die auf Text in der Zwischenablage zugreifen können (z. B. Shell, ED und MEmacs).

Klicken Sie anschließend mit der Maus die Stelle an, an der der Text eingefügt werden soll. Drücken Sie zum Einfügen des Texts die Tastenkombination "Rechte Amiga-Taste"-V.

Hinweis Wird ein Textblock in ein Shell-Fenster eingefügt, wird von der Shell versucht, jede Textzeile wie einen Befehl auszuführen. Dies kann zu unvorhersehbaren Ergebnissen führen, wenn in den Text Steuercodes für Eingabetasten eingebettet sind und es sich nicht um ein AmigaDOS-Skript handelt.

2.4.4 Arbeiten mit der Shell

Die folgenden Hinweise dienen zur Beschleunigung Ihrer Arbeit mit der Shell.

- **Verwenden Sie die Befehlsvorgeschichte und edieren Sie Befehlszeilen.**

  Insbesondere in der Lernphase benötigen Sie bisweilen mehrere Versuche, um einen AmigaDOS-Befehl richtig einzugeben. Rufen Sie mit den Cursortasten eine zuvor eingegebene Befehlszeile ab und ändern Sie nur den fehlerhaften Teil der Befehlszeile. Dies erspart Ihnen die erneute Eingabe der vollständigen Befehlszeile.

- **Verwenden Sie Alias-Namen.**

  Definieren Sie kurze Alias-Namen für häufig verwendete Befehle, um zusätzlich Zeit einzusparen. Dadurch erübrigt sich darüber hinaus das Auswendiglernen langer und/oder komplexer Optionsfolgen. Detaillierte Anweisungen zum Definieren von Alias-Namen können Sie unter dem Befehl ALIAS in Kapitel 6 nachlesen.

- **Verzichten Sie auf überflüssige Schlüsselwörter.**

  Zur klareren Darstellung werden in diesem Handbuch häufig optionale AmigaDOS-Befehlsnamen und -Schlüsselwörter genannt. Beim Erlernen des Befehlsformats ist jedoch nur selten die Angabe optionaler Schlüsselwörter erforderlich.

- **Verwenden Sie Kleinbuchstaben.**

  Alle Buchstaben von Befehlsnamen, Schlüsselwörtern und zugeordneten Verzeichnissen werden in diesem Handbuch mit Großbuchstaben dargestellt, obwohl von AmigaDOS die Groß- und Kleinschreibung nicht berücksichtigt wird. Dies dient zur Unterscheidung zwischen Schlüsselwörten und Dateinamen oder anderen Daten in den Beispielbefehlszeilen. Es gibt jedoch keinen Grund, bei der Eingabe von Befehlen ausschließlich Großbuchstaben zu verwenden, wenn keine Datei bzw. kein Verzeichnis erstellt werden soll, deren/dessen Name mit Großbuchstaben erscheinen soll.

- **Arbeiten Sie mit dem implizierten CD-Befehl.**

  Hierbei können Sie das Befehlswort CD auslassen. Dies erspart Ihnen drei Tastenanschläge. Geben Sie hinter der Aufforderung zum Wechseln von Verzeichnissen nur den Verzeichnisnamen, den Pfad, den Doppelpunkt oder Schrägstriche ein. Weitere Informationen zum Wechseln von Verzeichnissen können Sie unter dem Befehl CD in Kapitel 6 nachlesen.

Kapitel 3
Arbeiten mit AmigaDOS

Von AmigaDOS werden Daten in derselben hierarchischen Struktur gespeichert wie bei der Workbench. Für AmigaDOS-Befehle gelten spezielle Regeln, die Sie bei der Erstellung von Skripts und Programmen befolgen müssen, die auf Ihrem Amiga ausgeführt werden sollen. Zur erfolgreichen Verwendung von AmigaDOS müssen Sie mit den Grundbegriffen des Dateisystems und den AmigaDOS-Befehlskonzepten vertraut sein. In diesem Kapitel werden folgende Themen behandelt:

- Verwalten von Dateien, Verzeichnissen und Disks
- Grundlagen der Befehlszeilen
- Befehlsarten
- Befehlsstruktur
- Sonderzeichen
- Aufrufen von Programmen
- Benutzerspezifisches Anpassen der AmigaDOS-Umgebung

In Kapitel 6 und 7 dieses Handbuchs werden die einzelnen Befehle detailliert erläutert.

3.1 Verwalten von Dateien, Verzeichnissen und Disks

Damit Sie über AmigaDOS auf Daten zugreifen können, müssen Sie die Position der Daten kennen. Auf einem Amiga werden alle Daten in einem System von Verzeichnissen und Dateien gespeichert. Von AmigaDOS wird dasselbe Dateisystem verwendet wie von der Workbench. Der Unterschied zwischen den beiden liegt in der Arbeitsmethode. Insbesondere werden keine Piktogramme zum

Bearbeiten von Dateien und Verzeichnissen verwendet. Dem *Workbench-Benutzerhandbuch* können Sie detaillierte Informationen zum Amiga-Dateisystem und zur Verwendung allgemeiner Befehle entnehmen. In diesem Abschnitt erhalten Sie Informationen zu folgenden grundlegenden Konzepten von AmigaDOS:

- Dateisystembegriffe
- Dateiverwaltung
- Namenskonventionen
- Schlüsselwörter

3.1.1 Dateisystembegriffe

Die folgende Liste enthält die Hauptelemente des AmigaDOS-Dateisystems:

Gerät Ein physisches Gerät, z. B. ein Diskettenlaufwerk, oder ein logisches Gerät , d. h. eine Software-Einheit (z. B. RAM: oder logischer Drucker PRT:).

Partition Eine Festplatte oder ein Teil einer Festplatte, der von AmigaDOS als selbständiges Gerät behandelt wird.

Datenträger Eine bestimmte Disk oder ein Teil einer Disk, der von AmigaDOS als selbständiges Gerät behandelt wird. Disketten und Festplattenpartitionen sind Datenträger.

Verzeichnis Verzeichnisse entsprechen Workbench-Schubladen.

Hauptverzeichnis Das oberste Verzeichnis in einem Dateisystem für einen bestimmten Datenträger (engl. Root), d. h. das Verzeichnis, das alle anderen Verzeichnisse enthält.

Unterverzeichnis Ein Verzeichnis, das in einem anderen Verzeichnis enthalten ist.

Datei Eine benannte Sammlung von Daten.

Pfad Eine Folge aus Geräte-, Verzeichnis- und Unterverzeichnisnamen, die zur eindeutigen Angabe einer Datei und deren Standort dient.

3.1.2 Dateiverwaltung

AmigaDOS speichert Daten im Dateisystem eines Geräts. Dieses Dateisystem ist in Verzeichnissen, Unterverzeichnissen und Dateien organisiert. Die Verzeichnisse und Dateien sind in einer hierarchischen Struktur angeordnet, die auch als Baum bezeichnet wird. Jeder Ast des Baumes ist ein Verzeichnis, das wiederum mehrere Unterverzeichnisse enthalten kann. Am Ende der Äste (Verzeichnisse) befinden sich (wie Blätter) die Dateien, falls die Verzeichnisse nicht leer sind. Abb. 3-1 zeigt einen Verzeichnisbaum.

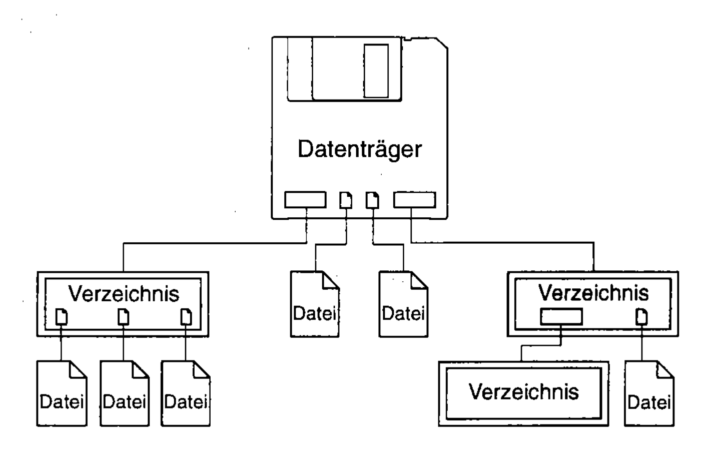


Abb. 3-1. Verzeichnisbaum (Beispiel)

3.1.2.1 Geräte

Bei Geräten (engl. Device) handelt es sich um logische Geräte und die Hardware, die an Ihrem Amiga angeschlossen ist, wie z. B. Disketten- und Festplattenlaufwerke, die RAM-Disk und Peripheriegeräte. Sie können mit verschiedenen Namen auf die Daten zugreifen, die auf diesen Geräten gespeichert sind:

Wenn Sie auf die Dateien eines bestimmten Datenträgers zugreifen wollen, können Sie mit Hilfe des zugehörigen Datenträgernamens, z. B. Workbench:, oder des zugehörigen Gerätenamens, z. B. DF0:, auf den gewünschten Datenträger verweisen. Sie können einen der

beiden genannten Namen verwenden. In beiden Fällen müssen Sie jedoch hinter dem Namen einen Doppelpunkt (:) eingeben. Wenn Sie mit dem Datenträgernamen auf eine Disk verweisen, durchsucht das System alle verfügbaren Laufwerke nach dem angegebenen Datenträger. Falls der angegebene Datenträger nicht gefunden werden kann, werden Sie zum Einlegen des gewünschten Datenträgers aufgefordert. Wenn Sie mit Hilfe des Gerätenamens auf eine Diskette verweisen, durchsucht das System nur den Datenträger, der sich gerade im angegebenen Gerät befindet.

AmigaDOS ordnet Peripheriegeräten, die über verschiedene Schnitstellen an den Computer angeschlossen sind, und logischen Geräten (Software-Einheiten) Standardnamen zu. Im allgemeinen dienen diese Geräte zur Ausgabe von Daten, z. B. zum Kopieren einer Datei auf einen Drucker. Diese Standardnamen können Sie der folgenden Liste entnehmen:

SYS: Steht für den Datenträger, den der Amiga nach den grundlegenden, auf Disk residierenden Ressourcen wie z. B. C: und LIBS: durchsucht.

PAR: Steht für ein Gerät, meist einen Drucker, das an der parallelen Schnittstelle angeschlossen ist. Wenn Sie eine Datei nach PAR: kopieren, wird sie an das Gerät gesendet, das an der parallelen Schnittstelle angeschlossen ist. Die auf diesem Weg gesendeten Ausgabedaten werden nicht durch Treibersoftware verändert.

SER: Steht für ein Gerät, das an der seriellen Schnittstelle angeschlossen ist, z. B. ein Drucker oder ein Modem. Die auf diesem Weg gesendeten Ausgabedaten werden nicht durch Treibersoftware verändert.

PRT: Bezeichnet den Drucker. Die nach PRT: gesendeten Daten durchlaufen den ausgewählten Druckertreiber und die serielle oder parallele Schnittstelle, je nachdem welche Angaben im Drucker-Voreinsteller (Printer) der Schublade "Prefs" (Voreinstellungen) gemacht wurden.

CON: Steht für eine Konsole mit einem Eingabefenster bzw. einem Textausgabefenster. Zu den Konsolenfenstern gehört u. a. das Shell-Fenster.

CONSOLE: Steht für das aktuelle Konsolenfenster. Anstelle von CONSOLE: kann auch ein Sternchen (*) verwendet werden.

NIL: Steht für ein Hilfsgerät, mit dem verhindert wird, daß Ausgabedaten am Bildschirm erscheinen. Alle an NIL: gesendeten Daten werden gelöscht.

RAM: Steht für die RAM-Disk, d. h. einem Teil des internen Amiga-Speichers, der als Speichergerät verwendet werden kann. Alle auf der RAM-Disk gespeicherten Daten gehen bei einem Neustart bzw. beim Ausschalten des Amiga verloren.

RAD: Bezeichnet eine spezielle Art von RAM-Disks, bei denen die gespeicherten Daten nur verloren gehen, wenn das System ausgeschaltet wird, aber nicht, wenn ein Neustart erfolgt. Weitere Informationen zu diesem Thema können Sie Anhang C entnehmen.

DF0: Steht für das interne Hauptdiskettenlaufwerk des Amiga, von dem aus der Amiga zu starten versucht, wenn kein anderes Gerät vorhanden ist, von dem aus das System gestartet werden kann.

3.1.2.2 Verzeichnisse

Amiga-DOS-Verzeichnisse entsprechen Workbench-Schubladen. Zusammengehörige Dateien können so in Verzeichnissen gruppiert und klassifiziert werden. Dabei befindet sich jede Datei einer Disk in einem Verzeichnis. Selbst leere, aber formatierte Disks enthalten ein Verzeichnis, das sog. Hauptverzeichnis (engl. Root). Wenn Sie eine Datei auf einer leeren Disk erstellen, befindet sich diese Datei im Hauptverzeichnis. Falls der Datei ein Piktogramm zugeordnet wurde, erscheint es im Disk-Fenster.

Verzeichnisse können neben Dateien auch andere Verzeichnisse, sog. "Unterverzeichnisse", enthalten. Der Amiga unterstützt eine nicht beschränkte Anzahl verschachtelter Verzeichnisse (Verzeichnisse in Verzeichnissen).

3.1.2.3 Dateien

Eine Datei, die grundlegende Speichereinheit eines Computers, ist eine strukturierte Sammlung von Daten. Alle Programme sowie alle permanent gespeicherten Daten, die von Programmen verwendet oder erzeugt werden, sind Dateien. Datendateien werden durch Projektpiktogramme dargestellt. Datendateien enthalten Daten, die

von einem Programm erzeugt oder verwendet werden (z. B. Text-, Grafik- oder Tabellenkalkulationsdateien).

3.1.2.4 .Info-Dateien

Neben den genannten Dateien werden vom Amiga auch sog. ".info-Dateien" verwendet. Die .info-Dateien enthalten die Piktogramme, die auf dem Workbench-Bildschirm angezeigt werden. Alle Dateien oder Verzeichnisse, die ein Piktogramm besitzen, verfügen über eine dazugehörige .info-Datei. In .info-Dateien werden neben den Grafik- und Positionsdaten des Piktogramms auch die Angaben gespeichert, die im Fenster "Information" des jeweiligen Piktogramms unter "Standardprogramm" und "Merkmale" gemacht wurden.

Bei der Arbeit in der Shell behandelt AmigaDOS die .info-Dateien nicht automatisch wie die zugehörigen Verzeichnisse und Dateien. Wenn Sie beispielsweise die Datei "Clock" (Uhr) aus dem Verzeichnis "Utilities" (Hilfsprogramme) in das Systemverzeichnis kopieren (Befehl COPY), wird die Datei "Clock.info" im Gegensatz zur Workbench (Ziehen einer Schublade in eine andere Schublade) nicht automatisch mitkopiert. Wenn das Uhrenpiktogramm nach einem Kopiervorgang mit AmigaDOS in der Systemschublade erscheinen soll, müssen Sie die Datei "Clock.info" separat kopieren.

Wenn Sie Piktogrammbilder durch Kopieren von .info-Dateien ändern, achten Sie darauf, daß Sie ein Piktogramm desselben Typs kopieren: Programm (Dienstprogramm), Projekt, Schublade, Disk oder Papierkorb. Wenn der Piktogrammtyp nicht mit dem Typ der zugehörigen Datei übereinstimmt, können Sie es ggf. nicht von der Workbench aus öffnen. Der Piktogrammtyp wird im Informationsfenster für das jeweilige Piktogramm angezeigt, und Sie können ihn mit Hilfe des Programms IconEdit (Programm zum Edieren von Piktogrammen) ändern.

Jede Disk besitzt eine zugehörige Disk.info-Datei. Falls Sie diese Disk.info-Datei löschen, wird das vorherige Piktogramm automatisch durch ein vorgegebenes Disk-Piktogramm ersetzt.

3.1.3 Namenskonventionen

Für Datei- und Verzeichnisnamen gelten folgende Namenskonventionen:

- Namen können aus maximal 30 Zeichen bestehen und sowohl Groß- und Kleinbuchstaben als auch Satzzeichen ohne Sonderfunktion enthalten. Namen für Workbench-Dateien und -Schubladen dürfen maximal 25 Zeichen lang sein, damit eine mögliche .info-Erweiterung in den Dateinamen aufgenommen werden kann.
- Aufgrund ihrer Sonderfunktion dürfen Doppelpunkte (:) und Schrägstriche (/) nicht in Datei- und Verzeichnisnamen verwendet werden. Die folgenden Zeichen haben zwar keine Sonderfunktion, aber dennoch ist von deren Verwendung in Datei- und Verzeichnisnamen abzuraten, da sie in AmigaDOS eine besondere Bedeutung haben: Semikola (;), Sternchen (*), runde Klammern (( )), Fragezeichen (?), umgekehrte Apostrophe (`), Nummern- oder Pfundzeichen (#), eckige Klammern ([ ]), spitze Klammern (< >), Tilden (~), vertikale Striche (|), Dollarzeichen ($), doppelte Anführungszeichen (") und Prozentzeichen (%).
- Großbuchstaben in Dateinamen werden zwar beibehalten, aber die Groß- und Kleinschreibung wird von AmigaDOS nicht berücksichtigt. Dies bedeutet, daß AmigaDOS den Dateinamen allein anhand der Zeichenkette identifiziert. "TextDatei" ist für AmigaDOS dasselbe wie "textdatei".
- Bei der Arbeit mit AmigaDOS können Sie zwar Leerzeichen in Namen verwenden, aber es empfiehlt sich, darauf zu verzichten. Wenn Sie dennoch Namen mit Leerzeichen verwenden, müssen Sie den vollständigen Pfad in doppelte Anführungszeichen setzen. Es ist vorteilhafter, wenn Sie Unterstreichungszeichen (_) als Trennzeichen verwenden.

Hinweis Achten Sie besonders darauf, nie ein Leerzeichen am Anfang oder Ende eines Namens einzugeben. Diese Leerzeichen werden bei deren Anzeige nicht besonders hervorgehoben und leicht übersehen. Sie müssen jedoch eingegeben werden, damit AmigaDOS die Namen erkennt.

3.1.4 Schlüsselwörter

Schlüsselwörter sind Wörter, die von einem AmigaDOS-Befehl als Argument oder Option erkannt werden. Falls ein Konflikt zwischen einem Namen und einem Befehlsschlüsselwort auftritt, setzen Sie den Namen in Anführungszeichen, damit er als Name erkannt wird. Auf Ihrem System befindet sich beispielsweise ein Verzeichnis mit dem Namen "Files", und es sollen Informationen zu allen zugehörigen Dateien und Unterverzeichnissen abgefragt werden. Dazu können Sie den Befehl **LIST Files** verwenden. Dieser Befehl ist jedoch nicht eindeutig, da der Befehl LIST das Schlüsselwort Files besitzt. Diesen Konflikt umgehen Sie mit der folgenden Eingabe:

```
LIST "Files"
```

3.2 Grundlagen der Befehlszeilen

Zur effektiven Nutzung der Befehlszeileneingabe (CLI), z. B. der Amiga-Shell, müssen Sie mit den grundlegenen Konzepten dieser einzigartigen Methode zur Verwendung Ihres Computers vertraut sein. Dazu gehören u. a. folgende Konzepte:

- Unterscheidung zwischen Dateien, Programmen, Befehlen und Skripts
- Aktuelles Verzeichnis
- Befehlsstruktur

3.2.1 Dateien, Programme, Befehle und Skripts

Dateien, Programme, Befehle und Skripts sind benannte Sammlungen von Daten, die sich im Speicher Ihres Computers oder auf einer Disk befinden können. Diese Begriffe können sehr verwirrend sein, da sich deren Bedeutung häufig überschneidet.

3.2.1.1 Dateien

Bei Programmen, Befehlen und Skripts handelt es sich um Dateien. Dateien können an einer beliebigen Stelle auf einer Disk oder im Speicher des Amiga gespeichert sein. Dabei ist jedoch zu beachten, daß bestimmte Arten von Dateien in der Regel an spezifischen Standorten gespeichert werden.

3.2.1.2 Programme

Ein Programm ist eine Datei, die vom Computer zur Ausführung einer Aufgabe ausgeführt wird. Bei der Software, die Sie für Ihren Amiga erwerben, handelt es sich meistens um Programme. Workbench-Programme werden als Hilfsprogramme (Hilfsmittel), Dienstprogramme oder Editoren bezeichnet. Eine Datendatei, die Daten enthält, die ein Programm verwenden kann, z. B. Text oder Grafiken, ist der typische Fall einer Datei, die kein Programm enthält. Programme können an einer beliebigen Stelle gespeichert werden.

3.2.1.3 Befehle

Ein Befehl ist eine Art Programm. Der Begriff Befehl bezieht sich in der Regel auf Programme, die mit Hilfe einer Befehlszeile, z. B. der Shell, aufgerufen werden. Insbesondere sind damit Programme gemeint, die zusammen mit dem Computer geliefert werden und die als Teil des Betriebssystems zur Ausführung grundlegender Funktionen dienen. Bei den in Kapitel 6 dieses Handbuchs beschriebenen Programmen handelt es sich um AmigaDOS-Befehle. AmigaDOS-Befehle, bei denen es sich nicht um interne (in die Shell integrierte Befehle) handelt, werden im Verzeichnis C: gespeichert.

Der Begriff Befehl kann sich auch auf ein spezifisches Aufrufen des Programms, ggf. mit zugehörigen Argumenten, beziehen. Das Aufrufen des Programms für einen bestimmten Befehl wird in diesem Handbuch als Befehlszeile bezeichnet. Beispiel: "Die Befehlszeile `TYPE S:User-startup` ist ein Beispiel des Befehls TYPE." Der Befehl muß stets als erstes in der Befehlszeile stehen.

3.2.1.4 Skripts

Ein Skript ist eine weitere Art von Programmen, die eine Reihe von Befehlen enthält, aus denen sich das Programm zusammensetzt. Mit Hilfe eines Texteditors können Sie Skripts laden und edieren. Skripts dienen in der Regel zur Ausführung einfacher Aufgaben, die durch Edieren des Skripts geändert werden können.

In diesem Handbuch bezieht sich der Begriff Skript auf Dateien mit AmigaDOS-Befehlen. AmigaDOS-Skripts werden in der Regel im Verzeichnis S: gespeichert. ARexx-Programme werden ebenfalls als Skripts bezeichnet, obwohl sie auch Makros oder Programme genannt werden. Diese Skripts werden unter Verwendung der Zuordnung REXX: auch im Verzeichnis S: gespeichert. Bei einigen Computersystemen werden Skripts als Stapeldateien bezeichnet.

3.2.2 Suchpfad

Bei Verwendung der Shell muß auf dem Amiga bekannt sein, an welcher Stelle nach den auszuführenden Befehlen gesucht werden soll. Die Shell verfügt über einen Suchpfad, der Ihnen die Eingabe von Befehlen ohne Angabe des vollständigen Pfads ermöglicht. Beim Suchpfad handelt es sich um eine Folge von mehreren Verzeichnissen, die AmigaDOS durchsucht, wenn Befehle ohne Pfadangaben eingegeben werden.

Der Standardsuchpfad umfaßt das Verzeichnis C:, das aktuelle Verzeichnis und einige andere Verzeichnisse, die in der standardmäßigen Skript-Datei "Startup-sequence" (Startdatei) aufgeführt sind. Mit Hilfe des Befehls PATH (Pfad) oder unter Verwendung mehrfacher Zuordnungen mit Hilfe des Befehls ASSIGN (Zuordnen) können Sie weitere Verzeichnisse hinzufügen, in denen häufig verwendete Programme enthalten sind. Diese beiden Methoden weisen jedoch einen wichtigen Unterschied auf. Verzeichnisse, die mit Hilfe

des Befehls PATH hinzugefügt werden, sind lokal auf die Shell, in der sie hinzugefügt wurden, und auf die von dieser Shell aufgerufenen, untergeordneten Shells begrenzt. Verzeichnisse, die durch eine Mehrfachzuordnung mit Hilfe des Befehls ASSIGN hinzugefügt wurden, gelten für das gesamte System.

Wenn Sie Daten in der Shell eingeben, werden von AmigaDOS alle Verzeichnisse im Suchpfad nach einem Befehl mit diesem Namen durchsucht. Die Verzeichnisse werden in der Reihenfolge durchsucht, in der sie im Suchpfad genannt sind. Die Suche wird beendet, sobald der Befehl gefunden wird oder das Ende der Pfadliste erreicht ist. Wird ein Befehl in keinem der im Suchpfad angegebenen Verzeichnisse gefunden, erscheint die Meldung **`Unbekannter Befehl`** (siehe Abb. 3-2).

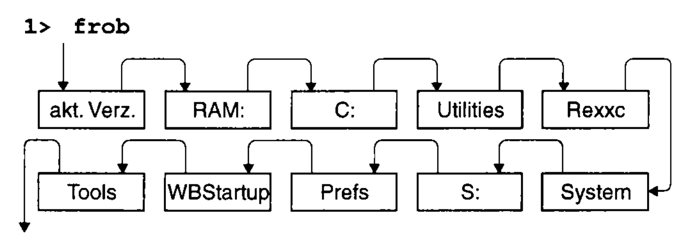


Abb. 3-2. Suchpfad

Hinweis — AmigaDOS verwendet den Suchpfad nur zur Suche von Befehlen. Wird in einem Befehlsargument auf eine Datei verwiesen, müssen Sie dort immer den vollständigen Pfad eingeben, wenn sie nicht im aktuellen Verzeichnis liegt.

3.3.3 Aktuelles Verzeichnis

Das aktuelle Verzeichnis ist die aktuelle Position der Shell in der Dateisystemhierarchie. Dies entspricht dem aktuellen Fenster in der Workbench. Der Name des aktuellen Verzeichnisses wird in der vorgegebenen Shell-Eingabeaufforderung angezeigt, so daß Sie jederzeit wissen, in welchem Verzeichnis Sie gerade arbeiten. Die folgenden Eigenschaften gelten für das aktuelle Verzeichnis

- Jede Shell verfügt über ein eigenes, unabhängiges aktuelles Verzeichnis.
- Jede Shell verfügt ausschließlich über ein aktuelles Verzeichnis.
- Das aktuelle Verzeichnis ist stets das erste Verzeichnis im Suchpfad.
- Der Pfad bis zum aktuellen Verzeichnis (einschließlich) wird stets als bekannt vorausgesetzt und muß für Befehle, die sich im aktuellen Verzeichnis befinden, nicht eingegeben werden.
- Das aktuelle Verzeichnis ist das Standardverzeichnis, für das ein Befehl ausgeführt wird, wenn kein anderes Verzeichnis angegeben wurde.

Durch Wechseln des aktuellen Verzeichnisses und durch Hinzufügen von Verzeichnissen zum Suchpfad reduziert sich die Textmenge, die zur Eingabe von Befehlszeilen einzutippen ist. Sie müssen häufig mehrere Operationen in einem bestimmten Verzeichnis ausführen, z. B. Kopieren, Umbenennen und Löschen von Dateien. In diesem Fall können Sie sich die Eingabe des vollständigen Pfads für jede Datei ersparen, indem Sie das Verzeichnis, in dem sich der Großteil der Dateien befindet, zum aktuellen Verzeichnis machen.

3.4 Befehlsarten

AmigaDOS-Befehle werden in "auf Disk basierende" und "interne" Befehle unterschieden.

Auf Disk basierende Befehle müssen vor deren Ausführung von der Disk geladen werden. Bei Systemen mit Festplatte kann jederzeit auf Disk basierende Befehle zugegriffen werden, da die Befehle bei deren Aufruf automatisch geladen werden. Bei Systemen ohne Festplatte

werden diese Befehle von einer Diskette gelesen, die ggf. vor Aufrufen des Befehls in ein Diskettenlaufwerk eingelegt werden muß.

Interne Befehle befinden sich in der Shell, d. h. im ROM (Read Only Memory - Nur-Lese-Speicher). Vom System wird direkt auf interne Befehle zugegriffen.

Einige AmigaDOS-Befehle entsprechen im wesentlichen Menüpunkten oder Programmen der Workbench. Diese Befehle und deren Workbench-Entsprechungen sind in der folgenden Tabelle aufgeführt:

Befehl	Funktion	Workbench-Entsprechung
CD	Aktuelles Verzeichnis wechseln	Anderes Fenster/Piktogramm auswählen
COPY	Datei, Verzeichnis oder Disk kopieren	Menüpunkt "Kopieren", mit Maus ziehen
DATE	Richtige/s Datum und Uhrzeit einstellen	Zeit-Voreinsteller (Prefs/Time)
DELETE	Datei oder Verzeichnis löschen	Menüpunkt "Löschen"
DIR	Alle Dateien eines Verzeichnisses anzeigen	Menüpunkt "Inhalt anzeigen alle Dateien"
DISKCOPY	Disk kopieren	Menüpunkt "Kopieren"
ENDSHELL	Shell-Fenster schließen	Schließsymbol eines Shell-Fensters auswählen
FORMAT	Disk formatieren	Menüpunkt "Disk formatieren"
INFO	Informationen zu allen Disks anzeigen	Titelleisten der Disk-Fenster beachten
LIST	Dateien mit Größenangaben usw. anzeigen	Menüpunkt "Inhalt auflisten nach Namen"

Befehl (Forts.)	Funktion (Forts.)	Workbench-Entsprechung (Forts.)
MAKEDIR	Neues Verzeichnis erstellen	Menüpunkt "neue Schublade"
NEWSHELL	Neues Shell-Fenster öffnen	Shell-Piktogramm öffnen
RELABEL	Disk-Datenträger auf angegebenem Laufwerk auf angegebenen Namen umbenennen.	Menüpunkt "umbenennen"
RENAME	Datei oder Verzeichnis umbenennen	Menüpunkt "umbenennen"
SETCLOCK	Datum und Uhrzeit speichern	Zeit-Voreinsteller (Prefs/Time)
TYPE	Inhalt einer Textdatei anzeigen	Programm MultiView

3.5 AmigaDOS-Befehlsstruktur

Jeder AmigaDOS-Befehl verfügt über ein spezifisches Format und eine bestimmte Syntax, die eingehalten werden muß, damit das System den Befehl akzeptiert und ausführt. Für die Arbeit mit AmigaDOS-Befehlen gibt es nur wenige Regeln, aber diese sind strikt einzuhalten:

- Ein gültiger Befehl oder Programmname muß als erstes in der Befehlszeile stehen. Der vollständige Pfad zum Befehl ist nicht erforderlich, wenn der Befehl sich in einem Verzeichnis beindet, das im Suchpfad angegeben ist.
- Argumente werden durch Leerzeichen vom Befehl und voneinnder getrennt. Dabei reicht ein Leerzeichen, aber zusätzliche Leerzeichen sind zulässig. Neben den befehlsspezifischen Satzeichen dürfen keine weiteren Satzzeichen verwendet werden.
- Von AmigaDOS wird die Groß- und Kleinschreibung nicht beücksichtigt. Beliebige, in die Befehlszeile eingegebene Zeichenketten, die sich nur durch Groß- und Kleinschreibung unter-

scheiden, werden identisch verarbeitet. Die Groß- und Kleinbuchstaben in Datei- und Verzeichnisnamen bleiben erhalten.

- Falls nicht anders angegeben, muß der gesamte Pfad bzw. die vollständige Zeichenkette in doppelte Anführungszeichen (") gesetzt werden, wenn ein Pfad- oder Zeichenkettenargument ein Leerzeichen enthält. Beispiel:

```
1> ECHO Kommentar TO Adisk:Text/Kommentar
1> ECHO "Ein Kommentar" TO "Disk 4:Text/Kommentar"
```

- Eine Standard-Shell-Befehlszeile kann aus maximal 512 Zeichen bestehen.

In Abb. 3-3. ist ein Beispiel für die Struktur eines AmigaDOS-Befehls dargestellt. Diese Befehlszeile setzt sich aus dem Befehl COPY und zwei nachgestellten Argumenten zusammen.

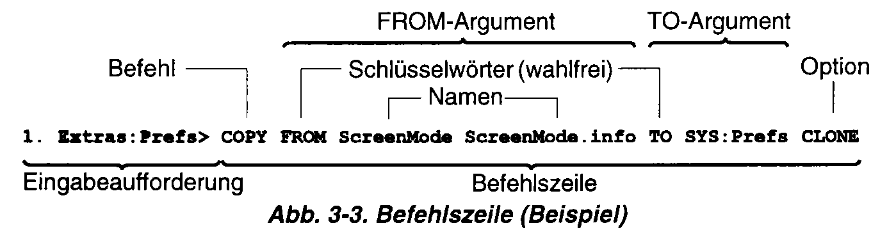


Abb. 3-3. Befehlszeile (Beispiel)

Ein Argument ist eine zusätzliche Information, die vom Befehl verwendet wird, z. B. ein Dateiname oder eine Option. Befehlsargumente sind mit den Merkmalen von Piktogrammen der Workbench vergleichbar. Je nach Befehl sind Argumente optional oder erforderlich. Abb. 3-3 zeigt folgende Punkte zu Argumenten:

- Schlüsselwörter für AmigaDOS-Befehle sind in der Regel vollständige Wörter oder einfache Abkürzungen, z. B. FROM (Von) und TO (Zu).
- Ein Argument kann aus mehr als einer Komponente bestehen. Im Beispiel enthält das Dateinamenargument gleichzeitig zwei Namen: Screenmode und Screenmode.info.
- Einige Argumente sind durch ein Schlüsselwort gekennzeichnet. Dabei kann es sich um optionale oder erforderliche Schlüsselwörter handeln.
- Werden optionale Schlüsselwörter in einem Befehl mit mehreren Argumenten ausgelassen, müssen die verbleibenden Argumente

in der Reihenfolge eingegeben werden, die in der zugehörigen Befehlsschablone erscheint.

3.6 AmigaDOS-Sonderzeichen

Einige Zeichen erfüllen spezielle Funktionen, wenn sie in Verbindung mit AmigaDOS verwendet werden. Diese Sonderzeichen sind für die nachstehenden Funktionen wichtig:

- Angeben von Pfaden
- Namensmuster
- Umleiten von Befehlseingabe und -ausgabe

Bei Verwendung von AmigaDOS ist es wichtig, mit den möglichen Sonderfunktionen von Zeichen vertraut zu sein. Beachten Sie, daß ein Zeichen in unterschiedlichen Kontexten unterschiedliche Sonderfunktionen oder gar keine Sonderfunktion haben kann. Falls Sie einen Befehl eingeben, der korrekt zu sein scheint, aber zu unerwarteten Ergebnissen führt, überprüfen Sie, ob ein Zeichen des Befehls eine Sonderfunktion hat.

3.6.1 Befehlszeilenzeichen

Der Doppelpunkt und Schrägstriche sind bei AmigaDOS für die Angabe von Pfaden reserviert. In Dateiauswahlfenstern, in der Befehlszeile und in Skripts dienen diese Zeichen ausschließlich als Trennzeichen zwischen den einzelnen Komponenten der Pfadangabe.

Doppelpunkt (:)

Der Doppelpunkt wird zur Kennzeichnung von Gerätenamen (DF0:), Datenträgernamen (Workbench:) und zugeordneten Verzeichnissen (SYS:) verwendet. Vor dem Doppelpunkt darf kein Leerzeichen stehen, sofern er nicht das erste Zeichen der Befehlszeile ist. Außerdem darf ein Doppelpunkt und ein darauffolgender Datei- und Verzeichnisname im selben Pfad nicht durch ein Leerzeichen getrennt sein. Der Doppelpunkt allein steht für das Hauptverzeichnis des aktuellen Datenträgers. In den folgenden Beispielen wird der Doppelpunkt ordnungsgemäß verwendet:

```
1> DIR DEVS:
1> DIR DF0:Pictures
1> DIR :Prefs
1> DIR :
```

Schrägstrich (/)

Schrägstriche werden zur Trennung von Verzeichnis- und Dateinamen in einem Pfad verwendet. Beispiel:

```
1> LIST Berichte/Vertrieb/Ost
```

Mit den Schrägstrichen werden die drei Verzeichnisebenen voneinander getrennt. Bei diesem Beispiel werden die Dateien im Verzeichnis "Ost" aufgelistet.

Durch Eingabe eines einzelnen Schrägstrichs können Sie auch in der Verzeichnisstruktur um eine Ebene nach oben wechseln. Beispiel: Der aktuelle Pfad lautet "Berichte/Vertrieb/Ost", und Sie geben folgendes ein:

```
1> CD /
```

Dadurch wird in das Verzeichnis "Berichte/Vertrieb" gewechselt. Mit zwei Schrägstrichen können Sie um zwei Verzeichnisebenen nach oben wechseln usw.

Doppelte Anführungszeichen (")

Das doppelte Anführungszeichen selbst hat keine besondere Bedeutung. Bisweilen müssen Sie jedoch ggf. ein Befehlsargument in doppelte Anführungszeichen setzen, damit der Befehl ordnungsgemäß ausgeführt wird. Da Leerzeichen von AmigaDOS als Trennzeichen zwischen Argumenten verwendet werden, müssen Sie ein Argument, das Leerzeichen enthält (z. B. ein Pfad), in doppelte Anführungszeichen setzen, damit die Shell die einzelnen Teile des Arguments nicht als separate Argumente interpretiert. Folgender Befehl ist beispielsweise inkorrekt:

```
1> COPY Ram Disk:foo TO SYS:Anderdir
```

Es erscheint eine Fehlermeldung, da die Pfadangabe ein Leerzeichen enthält. Der Befehl gibt an, daß zwei Elemente kopiert werden sollen, obwohl es sich nur um ein einziges Element handelt. Wenn Sie den Pfad in doppelte Anführungszeichen setzen, wird er als ein einziges Argument interpretiert:

```
1> COPY "Ram Disk:foo" TO SYS:Anderdir
```

Zwei doppelte Anführungszeichen hintereinander sind der einfachste Verweis auf das aktuelle Verzeichnis. Beispiel:

```
1> COPY DF0:public.library TO ""
```

Ist das aktuelle Verzeichnis LIBS:, wird die Datei "public.library" in dieses Verzeichnis kopiert.

Pluszeichen (+)

Das Pluszeichen in Verbindung mit dem Befehl RUN verknüpft mehrere Befehle, die in aufeinanderfolgenden Zeilen eingegeben werden, zu einer einzigen Befehlszeile. Weitere Informationen und Beispiele finden Sie unter dem Befehl RUN in Kapitel 6.

Fragezeichen (?)

Das Fragezeichen dient u. a. zur Anzeige der Schablone eines Befehls. Die Schablone dient als Online-Hilfsinformation über Befehlsargumente. Geben Sie zum Aufrufen der Schablone für einen bestimmten Befehl den Befehlsnamen gefolgt von einem Leerzeichen und dem Fragezeichen ohne weitere Argumente ein:

```
1> TYPE ?
FROM/A/M,TO/K,OPT/K,HEX/S,NUMBER/S:
```

Daraufhin wird von der Shell die Schablone angezeigt. Sie haben jetzt die Möglichkeit, die Argumente für den Befehl einzugeben, dessen Schablone Sie mit dem Fragezeichen aufgerufen haben. Geben Sie die Argumente für den Befehl hinter dem Doppelpunkt ein. Achten Sie darauf, daß Sie vor Drücken der Eingabetaste nur die erforderlichen Argumente und Schlüsselwörter eingeben.

3.6.2 Namensmuster

Mit Hilfe von Namensmustern können Sie mit einem einzigen Befehl mehrere Dateien oder Verzeichnisse gleichzeitig bearbeiten. In Befehlsargumenten können Sie spezielle Jokerzeichen verwenden, die für Zeichen in Dateinamen stehen. Sie haben z. B. die Möglichkeit, alle Dateien, die einen bestimmten Anfangsbuchstaben oder dieselbe Namenserweiterung haben oder sich im selben Verzeichnis befinden, mit einem einzigen Befehl zu kopieren oder umzubenennen.

3.6.2.1 Jokerzeichen

In der folgenden Liste sind die verschiedenen Jokerzeichen und die durch sie ersetzten Zeichen aufgeführt. In der Liste bedeutet ein <p>, daß ein Zeichen bzw. eine Zeichenkette, das/die unmittelbar neben dem Jokerzeichen steht, verglichen wird. Wenn die Sonderfunktion der Jokerzeichen unterdrückt werden soll, stellen Sie dem entsprechenden Zeichen einen Apostroph (') voran, d. h. '? steht für ? und ' ' (zwei aufeindanderfolgende Apostrophe) für '.

? — Steht für ein einzelnes Zeichen.

#<p> — Bedeutet, daß <p> null- bis n-mal vorkommt.

<p1>|<p2> — Stimmt überein, wenn <p1> oder <p2> übereinstimmt.

~<p> — Stimmt mit allem außer <p> überein.

(<p1><p2>...) — Runde Klammern gruppieren Elemente.

[<p>-<p>] — Eckige Klammern begrenzen einen Zeichenbereich.

% — Steht für die Null-Zeichenkette (keine Zeichen).

'<p> — Ist <p> Jokerzeichen, wird durch den Apostroph die Sonderfunktion unterdrückt.

Die folgenden Beispiele zeigen die möglichen Übereinstimmungen bei Eingabe der Zeichenkette in der linken Spalte

A?B — Name, der aus drei Zeichen besteht, mit A beginnt und auf B endet (z. B. AcB, AzB und a3b).

A#BC — Name, der mit A beginnt, auf C endet und dazwischen beliebig viele Buchstaben B enthält (z. B. AC, ABC, ABBC, ABBBC).

ABC#? — Name, der mit ABC beginnt und beliebig endet (z. B. ABCD, ABCDEF.info oder ABCXYZ).

#?XYZ — Name, der auf XYZ endet und beliebig beginnt (z. B. ABCXYZ und ABCDEFXYZ).

A(B|C)D — ABD oder ACD.

~ (XYZ) — Alle Namen außer XYZ.

~ (#?XYZ) — Alle Namen, die nicht auf XYZ enden.

A#(BC) — Name, der mit A beginnt und beliebig viele nachgestellte BC-Kombinationen enthält (z. B. ABC, ABCBC und ABCBCBC).

A(B|D|%)#C	ABC, ADC, AC (% ist die Null-Zeichenkette), ABCC, ADCC, ACCC usw.
[A-D]#?	Alle Namen, die mit A, B, C oder D beginnen.
#?XYZ'?	Alle Namen, die auf XYZ? enden.

Am häufigsten wird die Kombination #? verwendet, die mit allen Zeichen übereinstimmt. #? entspricht dem Jokerzeichen *, das von anderen Computersystemen verwendet wird. Sollen beispielsweise alle .info-Dateien im Verzeichnis "Bilder" gelöscht werden, geben Sie folgendes ein:

```
1> DELETE Bilder/#?.info
```

Achtung **Verwenden Sie #? mit großer Sorgfalt, damit Sie nicht versehentlich den gesamten Inhalt einer Disk löschen.**

3.6.3 Umleitung

Mit den Umleitungszeichen können Sie die Ein-/Ausgabe auf eine bestimmte Datei oder ein bestimmtes Gerät (z. B. Drucker, Modem oder logisches Gerät) umleiten. Wenn Sie in der Shell arbeiten, werden Befehle normalerweise über die Tastatur eingegeben und die Daten im aktuellen Shell-Fenster ausgegeben. Sie können die Ein- und Ausgabe jedoch mit Hilfe der linken und rechten spitzen Klammer und des Sternchens umleiten.

3.6.3.1 Spitze Klammern

Das Umleitungsargument besteht aus dem Zeichen < (Eingabe umleiten) oder > (Ausgabe umleiten) gefolgt von einem Datei- oder Gerätenamen. Der spitzen Klammer muß ein Leerzeichen vorangestellt werden; ein nachgestelltes Leerzeichen ist nicht erforderlich.

Je nach Befehlssyntax können die Umleitungszeichen bei einigen Befehlen anstelle der Argumente TO und FROM verwendet werden.

Die Umleitung der Ein- bzw. Ausgabe erfolgt nur für die Befehlszeile, in der die Umleitungszeichen aufgeführt sind. Für nachfolgende Befehle ohne Umleitungsargumente verwendet AmigaDOS wieder die standardmäßigen Ein- und Ausgabewege.

Rechte spitze Klammer (>)

Mit Hilfe der rechten spitzen Klammer wird die Konsolenausgabe auf das Ziel umgeleitet, auf das mit der Klammer verwiesen wird. Bei der Konsolenausgabe handelt es sich um den Text, der bei Ausführung eines Befehls im Shell-Fenster ausgegeben wird. Beispiel:

```
1> DIR >Testdatei DF0:
```

Bei diesem Befehl wird die Ausgabe (Verzeichnisliste von DF0:) in die Datei "Testdatei" des aktuellen Verzeichnisses umgeleitet. Ist die Datei "Testdatei" nicht vorhanden, wird sie erstellt und die Verzeichnisliste als ASCII-Text in ihr gespeichert. Die Verzeichnisliste erscheint nicht auf dem Bildschirm.

Nur die Konsolenausgabe des Befehls wird umgeleitet, nicht aber die Daten, die mit dem Befehl bearbeitet werden. Beispiel:

```
1> COPY >Protokoll Bilder TO Bildarchiv: ALL
```

Mit diesem Befehl werden alle Dateien aus dem Verzeichnis "Bilder" auf die Disk "Bildarchiv" kopiert, und die Namen der kopierten Dateien werden in die Datei "Protokoll" umgeleitet.

Linke spitze Klammer (<)

Soll anstelle der Tastatur eine Datei als Eingabequelle für einen Befehl dienen, verwenden Sie das Symbol <. In diesem Fall müssen Sie jedoch zusätzlich in der Befehlszeile ein Fragezeichen (?) als separates Argument eingeben. In diesem Fall fordert das Fragezeichen den Befehl auf, die Eingabedaten zu akzeptieren, d. h. es dient nicht als Jokerzeichen. Im folgenden Beispiel wird zunächst eine Datei erstellt und anschließend der Inhalt der Datei als Argument für einen Befehl verwendet:

```
1> ECHO morgen TO Datumdat
1> DATE ? <Datumdat
```

Mit dem Befehl ECHO wird eine Datei erstellt, die das Wort morgen enthält. Der Befehl DATE akzeptiert den Inhalt der Datei "Datum" (das Wort "morgen") in der gleichen Weise wie dieselbe Eingabe über die Tastatur. Dadurch wird das Systemdatum auf den folgenden Tag eingestellt.

Doppelte rechte spitze Klammern (>>)

Mit Hilfe doppelter rechter spitzer Klammern ohne Leerzeichen (>>) werden die Ausgabedaten umgeleitet und an den Inhalt einer bestehenden Datei angehängt. Beispiel:

```
1> Postscript >>Laser/Brief
```

Das Programm "Postscript" wird aufgerufen, und die ausgegebenen Daten werden an die Datei "Laser/Brief" angehängt.

Sternchen (*)

Ein Sternchen verweist auf das aktuelle Shell-Fenster. Zur Vermeidung von Verwechslungen mit anderen Funktionen des Sternchens empfiehlt sich jedoch die Verwendung des Gerätenamens CONSOLE:, der dem * entspricht. Das Sternchen kann als Argument FROM bzw. TO oder als Dateiname für die Umleitung (Eingabequelle oder Ausgabeziel) verwendet werden.

Durch Drücken der Tastenkombination Ctrl-\ wird die Standardquelle bzw. das Standardziel für die Ein-/Ausgabe wiederhergestellt. Beispiel:

```
1> COPY * TO Schirmdatei
```
oder
```
1> COPY CONSOLE: TO Schirmdatei
```

Nach Eingabe dieses Befehls werden alle in das aktuelle Fenster eingegebenen Daten in die Datei "Schirmdatei" kopiert, bis Sie die Tastenkombination Ctrl-\ drücken.

Die Tastenkombination Ctrl-\ dient auch zum Schließen eines Shell-Fensters. Achten Sie also darauf, daß Sie bei Beenden einer Umleitung die Tastenkombination nicht zweimal drücken. Ansonsten wird das Shell-Fenster geschlossen.

3.7 Aufrufen von Programmen

Die meisten Programme können sowohl von der Workbench als auch von der Shell aus aufgerufen werden. Soll ein Programm von der Shell aus gestartet werden, geben Sie in der Regel hinter der Shell-Eingabeaufforderung den Programmnamen ein. (Befindet sich die

Programmdatei nicht im aktuellen Suchpfad, müssen Sie zusätzlich den vollständigen Pfad zur Datei eingeben.) Dadurch wird AmigaDOS angewiesen, das Programm zu laden und auszuführen.

Bei den meisten Programmen können Sie nach dem Programmnamen noch zusätzliche Informationen angeben, wie z. B. den Namen einer zu ladenden Datei oder andere Startoptionen. Diese zusätzlichen Elemente werden als Argumente bezeichnet. Lesen Sie in der Dokumentation zum jeweiligen Programm nach, welche Argumente zulässig sind und wie sie einzugeben sind.

Beispiele:

```
1> MEmacs
```

Der Editor MEmacs wird geladen und gestartet. Fügen Sie wie folgt ein Argument hinzu:

```
1> MEmacs S:User-startup
```

MEmacs wir geladen und gestartet, wobei automatisch die Datei "User-startup" (Benutzerstart) aus dem Verzeichnis "S:" zum Edieren geladen wird.

```
1> CLOCK WIDTH 200 HEIGHT 100 SECONDS
```

Die Systemuhr wird mit der Fenstergröße 200 x 100 Bildpunkte geladen und die Option SECONDS (Sekunden) aktiviert.

Die Möglichkeit, Argumente zu übergeben, wird häufig zur Arbeitsersparnis zur Verfügung gestellt. Auf diese Weise können Sie Argumente, die sonst nur über mehrere Menüoperationen erreichbar sind, direkt in der Befehlszeile angeben. Bei vielen Programmen, besonders denjenigen, die nur von einer Shell aus gestartet werden können, ist es sogar erforderlich, daß Dateinamen und/oder andere Argumente zusammen mit dem Programmnamen in die Befehlszeile eingegeben werden.

3.7.1 Ausführen von Programmen im Hintergrund

Sie können einen Programmnamen auch als Argument des Befehls RUN eingeben. In diesem Fall wird das Programm im Hintergrund

geladen und ausgeführt. Sobald das Programm gestartet ist, wird wieder die Shell-Eingabeaufforderung angezeigt.

Geben Sie beispielsweise folgendes ein:

```
1> MEmacs
```

Daraufhin wird der Editor MEmacs geöffnet. Sie können aber erst wieder Befehle eingeben oder das Shell-Fenster schließen, wenn Sie MEmacs verlassen haben.

Wenn Sie dagegen den folgenden Befehl eingeben, wird der Editor MEmacs geöffnet, und die Shell-Eingabeaufforderung erscheint wieder, d. h. Sie können weitere Befehle eingeben:

```
1> RUN MEmacs
```

Wird ein Programm mit dem Befehl RUN aufgerufen, wird dem Programm eine eigene Prozeßnummer zugeordnet und eine Meldung mit der neuen Prozeßnummer angezeigt (z. B. `[CLI 2]`).

Alle vom Programm generierten Ausgaben werden im Shell-Fenster angezeigt, von dem aus das Programm gestartet wurde.

Solange Programme geladen sind, die vom Shell-Fenster aus aufgerufen wurden, können Sie dieses Fenster nicht schließen. Wenn Sie z. B. MEmacs über die Shell geöffnet haben, können Sie das Shell-Fenster erst nach Verlassen von MEmacs schließen. Dies können Sie durch Verwendung des Geräts NIL: vermeiden. In Kapitel 8 finden Sie ein Beispiel für diese Methode.

3.8 Benutzerspezifisches Anpassen Ihrer AmigaDOS-Umgebung

Im folgenden finden Sie einige Hinweise zur Anpassung der AmigaDOS-Umgebung an Ihre spezifischen Erfordernisse.

- **Passen Sie die Shell-Eingabeaufforderung Ihren Erfordernissen an.**

  Durch Änderung der Farbe der Eingabeaufforderung über entsprechende Escape-Sequenzen läßt sich beispielsweise die Eingabeaufforderung leichter vom Rest der Befehlszeile und den

Ausgabedaten der Befehle unterscheiden. Auf diese Weise können Sie die Prozeßnummer und das aktuelle Verzeichnis leichter verfolgen, die in der Regel Teil der Eingabeaufforderung sind. Nähere Informationen zur Änderung der Shell-Eingabeaufforderung können Sie unter dem Befehl PROMPT in Kapitel 6 und in einem Beispiel in Kapitel 8 nachlesen.

- **Legen Sie eine logische Verzeichnisstruktur an und verwenden Sie aussagekräftige Namen.**

  Da Sie für den Zugriff auf Daten deren Standort kennen müssen, empfiehlt es sich, die Daten Ihrer Disks und Verzeichnisse logisch zu organisieren und Namen zu verwenden, die Aufschluß über den Inhalt geben. Erstellen Sie jedoch nicht ohne guten Grund sehr stark verschachtelte Verzeichnisstrukturen.

- **Verzichten Sie in Namen auf Leerzeichen und Sonderzeichen.**

  Zeichen mit Sonderfunktionen in AmigaDOS, z. B. # und ~, sind zwar in Namen zulässig, können aber bei Verwendung in der Befehlszeile zu Problemen führen. Verwenden Sie zur Trennung der Wörter in einem Namen Punkte (.), Unterstreichungszeichen (_) oder Großbuchstaben anstelle von Leerzeichen. Beispiel: "Alte.Datei", "Alte_Datei" oder "AlteDatei" anstelle von "Alte Datei".

- **Geben Sie zusammengehörigen Dateien konsistente Namen.**

  Wenn Sie zusammengehörige Dateinamen mit einer gemeinsamen Erweiterung oder mit Folgenummern versehen, erleichtern Sie sich dadurch die Verwendung von Namensmustern bei der Bearbeitung solcher Dateien.

- **Verwenden Sie in Pfaden zugeordnete Gerätenamen.**

  Zugeordnete Gerätenamen ermöglichen Ihnen die Verwendung kurzer, eingängiger Namen anstelle von langen Pfaden. Sie können beispielsweise schneller `ENVARC:` eingeben als `SYS:Prefs/Env-archive`. Ordnen Sie (mit ASSIGN) häufig verwendeten Verzeichnissen oder stark verschachtelten Verzeichnissen einen sinnvollen, kurzen Namen zu.

- **Erweitern Sie den Suchpfad.**

  Wenn Sie einige Befehle oder Programme häufig aufrufen, fügen Sie die zugehörigen Verzeichnisse mit Hilfe von PATH oder ASSIGN dem Suchpfad hinzu, um den Zugriff zu vereinfachen.

- **Probieren Sie aus.**

  Die beste Methode zum Erlernen der Funktionsweise von AmigaDOS ist das Ausprobieren. Wenn Sie bei möglicherweise "zerstörenden" Befehlen (z. B. dem Befehl DELETE (Löschen) mit Namensmustern) große Sorgfalt walten lassen oder mit Kopien von Dateien in der RAM-Disk arbeiten, können Sie mit AmigaDOS frei experimentieren.

Kapitel 4
Editoren

Ein Texteditor oder ein Textverarbeitungsprogramm ist für das Erstellen und Bearbeiten (Edieren) von Textdateien und Skriptdateien erforderlich. Die Software der Amiga Workbench wird mit drei Texteditoren geliefert. Diese werden im vorliegenden Kapitel in der folgenden Reihenfolge beschrieben:

- ED
- MEmacs
- EDIT

Jeder Amiga-Editor kann separat zum Edieren von Skripts und Programmen von AmigaDOS eingesetzt werden; mit ED und MEmacs können solche Dateien erstellt werden. Wenn Sie sich bereits mit dem UNIX-Editor Emacs auskennen, bietet sich für Sie möglicherweise der Editor MEmacs an. Wenn Sie Dateien edieren müssen, die Binärcode enthalten oder solche Dateien, die so groß sind, daß sie nicht in Ihren Hauptspeicher passen, arbeiten Sie mit EDIT. Wenn Sie mit keinem dieser Editoren vertraut sind, empfiehlt sich die Verwendung von ED.

Jeder Editor besitzt die Basisfunktionen eines Textverarbeitungssystems, allerdings unterstützt keiner dieser Editoren Formatierungsoptionen für Schriftart und -stil, z. B. Kursivschrift, Seitennumerierung oder die Verwendung unterschiedlicher Schriftarten (Fonts). Wenn Sie diese Funktionen benötigen, können Sie Textverarbeitungssoftware anderer Hersteller erwerben, die die entsprechenden Funktionen für Ihren Amiga bietet.

4.1 ED

ED ist ein ASCII-Bildschirm-Texteditore, auf dessen Funktionen über Menüs und Funktionstasten zugegriffen wird. ED ist einfach zu handhaben und eignet sich zum Edieren von Skript- und Startsequenzdateien, MountLists sowie von anderen einfachen Dateien. Verwenden Sie eine Maus oder die Tastatur, um ED-Operationen auszuführen. Die Menüs von ED sind zwar vorprogrammiert, Sie können sie jedoch auch den jeweiligen Anforderungen individuell anpassen, wenn Sie einmal mit dem Programm vertraut sind.

Hinweis Mit ED können keine Dateien mit Binärcode bearbeitet werden. Verwenden Sie dafür statt dessen EDIT oder MEmacs.

Die unterste Zeile des ED-Fensters ist die Statuszeile. Hier werden Meldungen, Eingabeaufforderungen und Befehle angezeigt. Dort angezeigte Fehlermeldungen verbleiben dort so lange, bis Sie einen weiteren ED-Befehl eingeben. Abbildung 4-1 zeigt das ED-Fenster mit der Statuszeile.

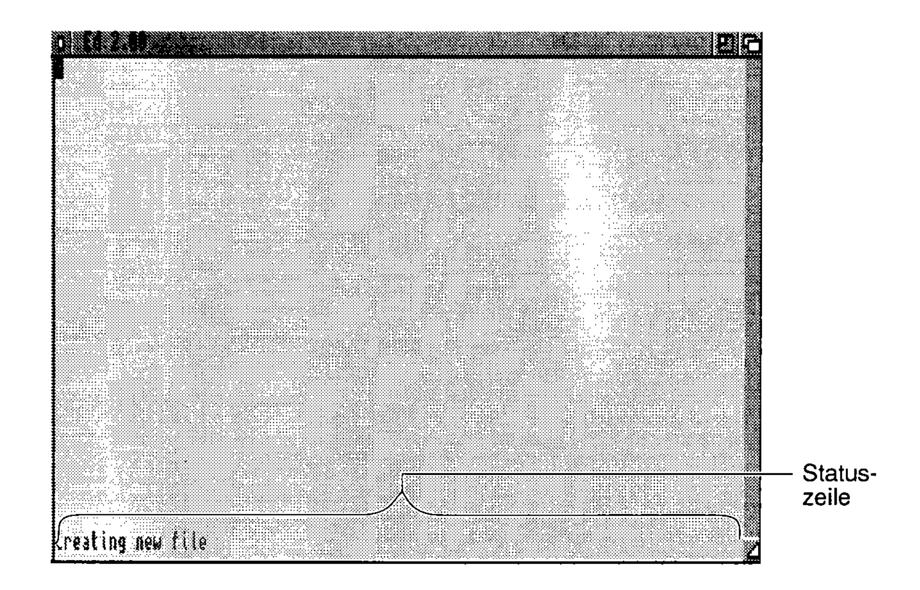


Abbildung 4-1. ED-Fenster mit Statuszeile

Der ED-Aufruf hat das folgende Format:

ED [FROM] <dateiname> [SIZE <n>] [*WITH* <dateiname>] [*WINDOW* <fensterspezifikation>] [TABS <n>] [WIDTH | COLS <n>] [HEIGHT | ROWS <n>]

Mit dem Argument FROM wird die zu edierende Quelldatei angegeben. Dieses Argument ist erforderlich, das Schlüsselwort FROM selbst dagegen ist wahlfrei.

Das Argument SIZE ändert die Größe des ED-Puffers. ED hat einen Textpuffer mit einer Standardgröße von 40.000 Byte. Beispiel:

```
1> ED Skript SIZE 55000
```

vergrößert den Puffer auf 55.000 Byte.

Das Argument WITH gibt eine ED-Befehlsdatei an, die eine beliebige Folge von ED-Befehlen des erweiterten Modus enthalten kann. Wenn WITH angegeben wurde, führt ED die in der Befehlsdatei stehenden Befehle aus. Das Schlüsselwort des Arguments WITH ist erforderlich, wenn WITH verwendet wird.

Mit dem Argument WINDOW legen Sie den Konsolentyp fest, z. B. RAW:0/0/640/256/EdFenster oder CONSOLE:. Das Schlüsselwort des Arguments WINDOW ist erforderlich, wenn WINDOW verwendet wird.

Mit TABS legen Sie den Tabulatorabstand fest, also die Anzahl Leerzeichen, die der Cursor nach rechts gerückt wird, wenn die Tabulatortaste gedrückt wird. Der Vorgabewert ist 3.

Mit den Argumenten WIDTH und HEIGHT stellen Sie die Größe des ED-Fensters ein. Dazu geben Sie die Anzahl Zeichen an, die horizontal und vertikal in eine Anzeige passen sollen. Standardmäßig hat das ED-Fenster die Größe 640 x 200 Pixel, das entspricht ca. 78 Zeichen in der Breite und 21 Zeilen Höhe.

4.1.1 Starten von ED

Starten Sie ED in einer Shell oder über den Workbench-Menüpunkt "Befehl ausführen". Öffnen Sie ED durch Eingabe von **ED** mit einem Namen einer neuen oder bereits vorhandenen Datei. Beispiel bei der Eingabeaufforderung:

```
1> ED <dateiname>
```

Dabei kann <dateiname> eine bestehende Datei oder eine neu zu erstellende Datei sein. Unter diesem Dateinamen werden die eingegebenen Daten gespeichert. Wenn die angegebene Datei im aktuellen Verzeichnis nicht gefunden wird, öffnet ED ein leeres Fenster, und es erscheint die Meldung **Creating new file** (Neue Datei wird erstellt).

4.1.2 Verwenden von ED

Alle ED-Befehle sind über Tastenfolgen, viele zusätzlich auch über Menüs aufrufbar. ED-Befehle können auf zwei Arten eingegeben werden:

1. Durch Auswahl des entsprechenden Menüpunkts.
2. Durch Eingabe der Befehlstastenfolge und Drücken der Eingabetaste.

Außerdem können manche Befehle mit Hilfe der Maus ausgeführt werden, z. B. zur Cursorbewegung.

Dateien können in ED in einem der folgenden Befehlsmodi bearbeitet werden:

Direkt Befehle werden unmittelbar nach Eingabe ausgeführt. ED wird im direkten Modus geöffnet.

Erweitert Befehle werden erst nach dem Drücken der Eingabetaste oder der Taste Esc ausgeführt.

4.1.2.1 Direkte Befehle

Im direkten Modus führt ED Befehle unmittelbar aus. Einen direkten Befehl geben Sie durch Drücken einer einzelnen Taste oder einer Ctrl-Tastenkombination oder mit Hilfe der Maus ein. Zu jedem direkten Befehl gibt es eine entsprechende erweiterte Version.

Direkte Befehle steuern folgende Aktionen:

- Cursorbewegung
- Durch Text blättern
- Text einfügen
- Text löschen
- Befehle wiederholen

4.1.2.2 Plazieren des Cursors im direkten Modus

Sie können den Cursor an jede Position im Text setzen, indem Sie den Mauszeiger an die entsprechende Stelle fahren und die Auswahltaste drücken. Über die Tastatur erfolgt die Cursorbewegung mit den Cursortasten, dem Tabulator und verschiedenen Ctrl-Tastenkombinationen.

Hinweis In ED wird mit der Tabulatortaste nur der Cursor bewegt. Es werden also keine Tabulatoren oder Leerzeichen in eine Zeile eingefügt.

Wenn Sie den Cursor um eine Position in eine der vier Richtungen weitersetzen wollen, drücken Sie die entsprechende Cursortaste. Befindet er sich am rechten Bildschirmrand, dann wird der Text nach

links gerollt, damit Sie die restliche Zeile sehen können. Der Text wird senkrecht um jeweils eine Zeile und waagrecht um jeweils 10 Zeichen gerollt. Der Cursor kann nicht über den oberen oder unteren Rand der Datei und über den linken Rand einer Zeile hinwegbewegt werden. In diesem Fall werden die Meldungen **`Top of File`** (Dateianfang) bzw. **`End of File`** (Dateiende) angezeigt.

Weitere Tastenkombinationen zur Steuerung des Cursors sind:

Shift-Cursor nach oben	Dateianfang
Shift-Cursor nach unten	Dateiende
Shift-Cursor nach links	Linker Rand des ED-Fensters (unabhängig von der Randeinstellung)
Shift-Cursor nach rechts	Ende der aktuellen Zeile
Ctrl-]	Ende der aktuellen Zeile (wenn sich der Cursor bereits dort befindet, wird er an den Zeilenanfang gestellt)
Ctrl-E	Anfang der ersten Zeile auf dem Bildschirm (wenn sich der Cursor bereits dort befindet, wird er ans Ende der letzten Zeile auf dem Bildschirm gestellt)
Ctrl-T	Anfang des nächsten Worts
Ctrl-R	Leerzeichen nach dem vorhergehenden Wort
Tab	Nächste Tabulatorposition (Vielfaches des TABS-Wertes - standardmäßig 3)

Wenn die Datei nicht auf einmal in das ED-Fenster paßt, können Sie durch die Datei blättern. Dies können Sie z. B., indem Sie den Cursor in die erste bzw. letzte Zeile des ED-Fensters stellen und die entsprechende Cursortaste drücken. Der Text wird dann um jeweils eine Zeile weitergeblättert. Wenn Sie mehrere Zeilen auf einmal weiterblättern wollen, verwenden Sie folgende Tastenkombinationen:

Ctrl-D	Blättert um 12 Zeilen nach unten.
Ctrl-U	Blättert um 12 Zeilen nach oben.

Durch diese Befehle wird die Position des Cursors im Fenster nicht verändert, d. h. nur der Text wird bewegt.

Wenn die Anzeige z. B. durch eine Meldung eines anderen Programms im ED-Fenster oder durch Meldungen in der Statuszeile beeinträchtigt wird, drücken Sie:

Ctrl-V	Damit wird die Fensteranzeige aktualisiert.

4.1.2.3 Einfügen von Text im direkten Modus

Im direkten Modus werden alle Tastatureingaben an der aktuellen Cursorposition eingefügt, und der Cursor wird entsprechend nach rechts gerückt. Die Zeichen rechts vom Cursor werden verschoben. Überschreitet die Zeilenlänge die Fensterbreite, so wird das Fenster nach links gerollt. Wenn Sie den Cursor mit der Tabulatortaste oder den Cursortasten über das Zeilenende hinweg plazieren, werden entsprechend viele Leerzeichen eingefügt.

Eine Zeile kann höchstens 255 Zeichen lang sein. Wenn Sie versuchen, mehr Zeichen einzugeben, erscheint die Meldung `Line Too Long` (Zeile zu lang).

Wenn Sie die aktuelle Zeile an der Cursorposition teilen wollen, drücken Sie die Eingabetaste. Text links vom Cursor verbleibt in der ursprünglichen Zeile, der Text unter dem Cursor und rechts davon wird nach unten in eine neue Zeile verschoben. Wenn der Cursor am Ende einer Zeile steht und Sie die Eingabetaste drücken, erscheint eine neue leere Zeile. In beiden Fällen erscheint der Cursor in der neuen Zeile in der ersten Spalte.

4.1.2.4 Löschen von Text im direkten Modus

In ED gibt es keinen Modus zum Überschreiben. Wenn Sie ein Wort oder eine Zeile ersetzen wollen, müssen Sie den betreffenden Text löschen und dann den neuen einfügen. Dazu stehen Ihnen einige Tasten und Tastenkombinationen zur Verfügung:.

Rücktaste	Löscht das Zeichen links vom Cursor.
Del	Löscht das Zeichen, auf dem der Cursor steht.
Ctrl-O	Befindet sich der Cursor auf einem Leerzeichen, werden alle Leerzeichen bis zum nächsten Zeichen gelöscht; befindet sich der Cursor auf einem anderen Zeichen, werden alle nachfolgenden Zeichen bis zum nächsten Leerzeichen gelöscht, also ein Wort.

Ctrl-Y Löscht alle Zeichen vom Cursor bis zum Zeilenende.

Wenn Sie Text löschen, verschieben sich alle Zeichen in der Zeile nach links, und Text, der über den rechten Fensterrand hinausging, wird sichtbar.

4.1.2.5 Vertauschen von Groß- und Kleinschreibung im direkten Modus

Sie können Groß- und Kleinschreibung vertauschen, indem Sie den Cursor an die gewünschte Stelle setzen und Ctrl-F drücken. Ein Kleinbuchstabe erscheint groß geschrieben und umgekehrt. Umlaute und Zeichen oder Symbole, die nicht im Alphabet enthalten sind, werden nicht geändert.

Wenn Sie Ctrl-F gedrückt haben, geht der Cursor um eine Stelle nach rechts. Ist das nächste Zeichen ein Buchstabe, so können Sie mit Ctrl-F dessen Groß- bzw. Kleinschreibung ändern. Dies können Sie so lange wiederholen, bis Sie alle Buchstaben einer Zeile geändert haben.

4.1.2.6 Erweiterte Befehle

Im erweiterten Modus werden Befehle in der Befehlszeile - oder Statuszeile - am unteren Rand des Fensters eingegeben. ED führt diese Befehle erst dann aus, wenn Sie die Eingabetaste oder Esc gedrückt haben. Wenn Sie erweiterte Befehle mit der Taste Esc abschicken, bleibt ED im erweiterten Modus. Schicken Sie die erweiterten Befehle dagegen mit der Eingabetaste ab, kehrt ED anschließend in den direkten Modus zurück.

Erweiterte Befehle haben folgende Aufgaben:

- Programmsteuerung
- Cursorbewegung
- Textbearbeitung
- Blockbefehle
- Suchen und Ersetzen von Text

Drücken Sie die Taste Esc, um in den Modus "erweiterte Befehle" zu schalten. In der Statuszeile erscheint als Eingabeaufforderung ein Stern. Erweiterte Befehle bestehen aus einem oder zwei Zeichen.

Groß- oder Kleinschrift ist dabei beliebig. Sie können mehrere Befehle in derselben Befehlszeile eingeben, wenn Sie diese durch ein Semikolon trennen. Befehle können in Gruppen zusammengefaßt werden, damit sie von ED automatisch wiederholt werden. Fehleingaben können Sie mit der Rücktaste korrigieren.

Befehle können auch über die programmierbaren Menüs und Funktionstasten ausgeführt werden. Konfigurieren Sie die Menüs und Funktionstasten um, indem Sie ihnen den jeweils gewünschten Befehl zuordnen (siehe dazu Kap. 4.1.5).

4.1.2.7 Verwenden von Begrenzungszeichen

In manchen Fällen müssen zu Befehlen auch Argumente angegeben werden, z. B. eine Zahl oder eine Textzeichenfolge. Ein Zeichenfolgenargument zu einem ED-Befehl muß zwischen zwei identischen Begrenzungszeichen stehen. In eindeutigen Situationen kann das nachgestellte Begrenzungszeichen weggelassen werden. Gültige Begrenzungszeichen sind ", /, \, !, :, +, - und %. Es ist nicht zulässig, das jeweils gleiche Begrenzungszeichen in der betreffenden Zeichenfolge zu verwenden. Buchstaben, Zahlen, Leerzeichen, Semikolons, Fragezeichen, Klammern und Steuerzeichen sind keine gültigen Begrenzungszeichen. Beispiel:

```
sa-RAM:Test
```

4.1.2.8 Verwenden eines Dateiauswahlfensters

ED kann auch aufgefordert werden, ein Dateiauswahlfenster zu verwenden, in welchem der Inhalt der Laufwerke und Verzeichnisse Ihres Systems angezeigt werden kann, wenn einem erweiterten Befehl ein Dateiname als Argument übergeben werden soll.

Um zum Laden oder Sichern einer Datei ein Dateiauswahlfenster aufzurufen, muß dem Zeichenfolgenargument des Befehls ein Fragezeichen (?) vorangestellt werden. Achten Sie darauf, daß vor dem Fragezeichen ein Leerzeichen steht (z. B. **`sa ?/Text/`**). Wenn auf einen Befehl eine Zeichenfolge folgt, betrachtet ED diese normalerweise als den Namen einer zu ladenden oder zu sichernden Datei und versucht, diese Operation unverzüglich auszuführen. Das Fragezeichen bedeutet aber, daß Sie die Datei über ein Dateiauswahlfenster angeben möchten. Nach dem Fragezeichen muß

auch dann noch eine Zeichenfolge stehen, diese Zeichenfolge erscheint dann aber in der Titelleiste des Dateiauswahlfensters.

4.1.3 ED-Menüs

ED besitzt zwei Gruppen von Menübefehlszuordnungen: Standard und Erweitert. Die Standardmenüzuordnungen (siehe Abbildung 4-2) werden über die Datei S:Ed-startup konfiguriert, die bei jedem Aufrufen von ED automatisch ausgeführt wird. Die Datei S:Ed-startup ist eine Befehlsdatei mit ED-Befehlen im erweiterten Modus, ohne Escape-Zeichen. Diese Datei kann zum Konfigurieren individuell angepaßter Menüs ediert werden (siehe Seite 4-24), oder Sie können vorprogrammierte Funktionstastenzuordnungen über den Menüpunkt Set FN Key definieren.

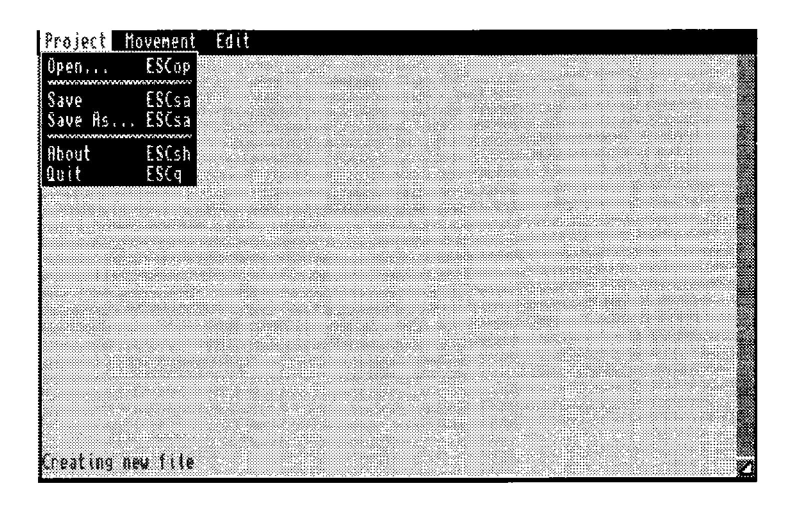


Abbildung 4-2. Standardmenüzuordnungen

4.1.3.1 Aktivieren erweiterter Menüs

Die Menüzuordnungen für erweiterte Befehle (siehe Abbildung 4-3) können durch Umbenennen oder Löschen der Standarddatei S:Ed-startup aktiviert werden. Wenn ED keine Datei mit Namen S:Ed-startup findet, startet das Programm unter Verwendung der erweiterten Menüs, die mehr Optionen bieten.

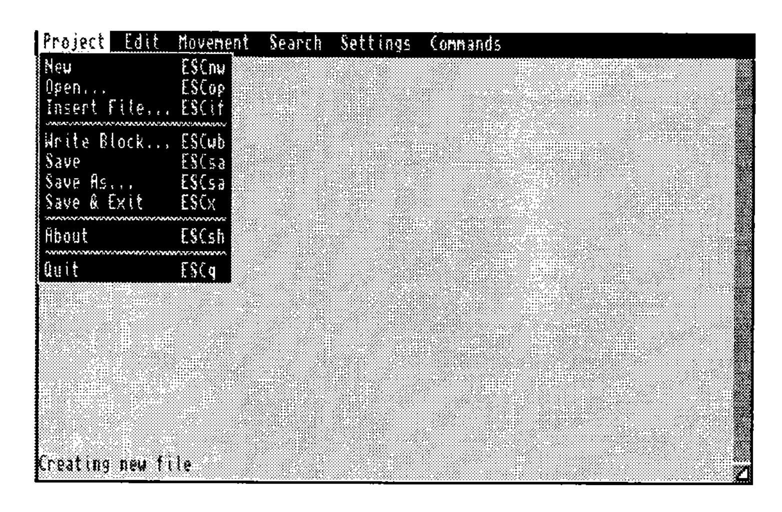


Abbildung 4-3. Erweiterte Menüzuordnungen

Falls Sie diese erweiterten Menüs benutzen wollen, empfiehlt es sich, die Datei S:Ed-startup nicht zu löschen, sondern wie folgt umzubenennen:

1. Gehen Sie im Workbench-Fenster in das Menü "Window" (Fenster) und wählen Sie "Show All Files" (Inhalt anzeigen alle Dateien).

2. Klicken Sie doppelt auf das Schubladenpiktogramm S.

3. Klicken Sie auf das Piktogramm Ed-startup.

4. Gehen Sie in das Menü "Icons" (Piktogramme) und wählen Sie "Rename" (Umbenennen).

5. Löschen Sie den Namen im Textfeld "New Name" (Neuer Name) des Dialogfensters "Rename" (Umbenennen) und geben Sie einen neuen Namen für Ed-startup ein.

6. Wählen Sie OK.

Sie können auch eine eigene, individuell angepaßte Datei mit Startoptionen erstellen. Vermeiden Sie dabei aber, Quit-Befehle in die Datei S:Ed-startup zu schreiben, da sonst ED sofort nach dem Öffnen wieder verlassen wird.

Die Menüpunkte in den Standardmenüs und den erweiterten Menüs haben unabhängig vom verwendeten Menüsatz die gleichen

Funktionen. Alle ED-Befehle sind mit erweiterten Befehlen über die Tastatur aufrufbar, selbst wenn sie in keinem Menü stehen.

In den folgenden Abschnitten werden die Menüpunkte der erweiterten Menüs beschrieben und die jeweils entsprechenden Befehle im erweiterten und direkten Modus angegeben. Eine Ellipse (...) bedeutet, daß ein Argument erforderlich ist oder daß ein Menüpunkt ein Dialogfenster oder eine Eingabeaufforderung aufruft. Die Buchstabenkombinationen der erweiterten Befehle lassen sich leichter merken, wenn man sie als Abkürzungen der Menütexte interpretiert.

4.1.3.2 Menü "Project"

Es folgen die Beschreibungen der Menüpunkte des erweiterten Menüs "Project":

New	Esc,N,W	Eine neue Datei wird erstellt und ersetzt die bestehende Datei. Die Meldung **`Edits will be lost-type Y to confirm:`** (Änderungen gehen verloren - mit Y bestätigen:) wird angezeigt. Der Befehl kann mit einer beliebigen Taste (außer Y) abgebrochen werden.
Open...	Esc,O,P...	Öffnet eine Datei. Geben Sie als Dateinamen den Pfad zu der Datei als korrekt abgegrenzte Zeichenfolge ein. (Wenn im Pfadnamen Schrägstriche vorkommen, darf der Schrägstrich nicht als Begrenzungszeichen benutzt werden.) Die Meldung **`Edits will be lost-type Y to confirm:`** (Änderungen gehen verloren - mit Y bestätigen:) erinnert Sie daran, daß Sie die aktuelle Datei ersetzen.
Insert File...	Esc,I,F...	Eine Datei wird in die aktuelle Datei eingefügt. ED lädt die angegebene Datei an die Stelle in den Speicher, die auf die aktuelle Zeile folgt.
Write Block...	Esc,W,B...	Schreibt den markierten Block in eine angegebene Datei. Eine eventuell vorhandene andere Datei mit demselben Namen wird überschrieben.

Save	Esc,S,A	Speichert den Text in der aktuellen Datei ab, wobei der eventuell vorher in dieser Datei vorhandene Text überschrieben wird. Mit Save As kann in eine andere Datei gespeichert werden. SA gefolgt von Q entspricht dem Befehl X.
Save As...	Esc,S,A...	Speichert den Text unter dem angegebenen Dateinamen ab.
Save & Exit	Esc,X	Verläßt ED und speichert die aktuelle Datei unter dem angegebenen Dateinamen. ED schreibt den Text aus dem Hauptspeicher in die Datei, die beim Öffnen von ED angegeben wurde; danach wird das Programm beendet.
About	Esc,S,H	Zeigt den aktuellen Status des Editors an. Es werden bestimmte Daten am Bildschirm angezeigt, z. B. die Werte der Tabulatoren, aktuelle Randeinstellungen, Blockmarkierungen und der Name der in Bearbeitung befindlichen Datei.
Quit	Esc,Q	Verläßt ED, ohne Änderungen zu speichern. Wenn zuvor in der Datei Änderungen vorgenommen wurden, erscheint eine Meldung mit der Frage, ob Sie ED trotzdem beenden wollen. Wenn Sie dies mit Y bestätigen, verlassen Sie ED, und die Änderungen gehen verloren.

4.1.3.3 Menü "Edit"

Es folgen die Befehle zum Edieren (Bearbeiten) von Dateien:

Undo Line	Esc,U	Macht in der aktuellen Zeile vorgenommene Änderungen rückgängig. Das Löschen einer ganzen Zeile kann ED allerdings nicht rückgängig machen. Sobald Sie die aktuelle Zeile verlassen haben, kann mit dem Befehl U keine Änderung in dieser Zeile mehr zurückgenommen werden.
Block Start **Block End**	Esc,B,S Esc,B,E	Bezeichnet Anfang und Ende eines Textblocks. Wenn Sie einen Block definieren möchten, der verschoben, eingefügt oder gelöscht werden soll, stellen Sie den Cursor in die erste Zeile des gewünschten Blocks und geben den Befehl BS ein. Gehen Sie dann mit dem Cursor in die letzte Zeile des gewünschten Blocks und geben Sie den Befehl BE ein. Ein Block kann nicht mitten in einer Zeile beginnen oder enden.
Show Block	Esc,S,B	Baut die Anzeige neu auf, so daß der Block oben in der Anzeige steht.
Insert Block	Esc,I,B	Fügt eine Kopie des Blocks hinter der aktuellen Zeile ein. Der Block bleibt so lange definiert, bis Sie seinen Text verändern. Verwenden Sie IB, um Kopien des Blocks mehrmals im ganzen Dokument einzufügen.
Delete Block	Esc,D,B	Löscht einen Block.
Delete Line	Esc,D Ctrl-B	Löscht die gesamte Zeile.

4.1.3.4 Menü "Movement"

Mit den folgenden Befehlen wird der Cursor in der Anzeige bewegt:

Top	Esc,T Shift-aufwärts	Dateianfang. Die erste Zeile der Datei wird zur ersten Zeile im Fenster.
Bottom	Esc,B Shift-abwärts	Dateiende. Die letzte Zeile der Datei wird zur letzten Zeile im Fenster.
Go To Line... (Move)	Esc,M...	Stellt den Cursor in die angegebene Zeile. Geben Sie die Zeilennummer in der Statuszeile ein und drücken Sie die Eingabetaste. Die angegebene Zeile wird zur obersten Zeile im Fenster. Wenn keine Zahl angegeben wird, springt der Cursor in die oberste Zeile des Fensters.
Next Page (Page Down)	Esc,P,D Ctrl-D	Weiter zur nächsten Seite.
Previous Page (Page Up)	Esc,P,U Ctrl-U	Zurück zur vorherigen Seite.

4.1.3.5 Menü "Search"

Die folgenden Befehle dienen zum Durchsuchen einer Datei nach bestimmten Textstellen. Es kann auch eine Suchzeichenfolge durch eine andere ersetzt werden (search and replace) und festgelegt werden, daß ED vor jedem Ersetzen eine Bestätigung anfordert. Wenn der angegebene Text nicht gefunden wird oder keine weiteren Vorkommen dieser Textzeichenfolge vorhanden sind, erscheint die Meldung **`Search failed`** (Suche gescheitert). Bei Verwendung der Menübefehle zum Suchen und Ersetzen (Find and Replace) fordert ED zur Eingabe der Textzeichenfolgen auf. Geben Sie den Text dann ohne Begrenzungszeichen ein. Im erweiterten Modus sind zu dem Befehl auch Begrenzungszeichen anzugeben.

Find... **Find Next** **(Search)**	Esc,s...	Sucht das nächste Vorkommen der angegebenen Textzeichenfolge. Die Suche beginnt mit dem Zeichen rechts von der aktuellen Cursorposition und geht in Vorwärtsrichtung bis zum Dateiende. Wenn die Zeichenfolge gefunden wird, springt der Cursor an den Anfang der gefundenen Zeichenfolge. Bei der Suche wird zwischen Groß- und Kleinschreibung unterschieden, es sei denn, der Befehl "Ignore Case" wird verwendet. "Find Next" wiederholt den Befehl.
Reverse Find... **Reverse Find Next** **(Back Find)**	Esc,B,F...	Durchsucht die Datei rückwärts nach der angegebenen Zeichenfolge. Mit diesem Befehl wird das jeweils letzte Vorkommen der gesuchten Zeichenfolge vor der aktuellen Cursorposition gefunden. Die Suche wird bis zum Dateianfang fortgesetzt. "Reverse Find Next" wiederholt den Befehl.
Replace... **(Exchange)**	Esc,E...	Ersetzt einen Text durch einen anderen. Im erweiterten Modus müssen die Zeichenfolgen mit drei Begrenzungszeichen eingegeben werden; wenn Sie z. B. das Wort 'mehr' durch 'Meer' ersetzen möchten, geben Sie `"mehr"Meer"` ein. Geben Sie zur Angabe leerer Zeichenfolgen nur zwei Begrenzungszeichen ohne Text ein. Wenn die erste Zeichenfolge leer ist, fügt ED die zweite an der aktuellen Cursorposition ein. Wenn die zweite Zeichenfolge leer ist, sucht ED nach dem nächsten Vorkommen der ersten und löscht diese. Beim Austauschen von Text werden Randeinstellungen nicht beachtet.
Global Replace...	Esc,R,P, E...	Ersetzt jedes Vorkommen des auszutauschenden Textes.
Query-Replace...	Esc,E,Q...	Sucht den auszutauschenden Text , fordert Sie jedoch vor dem Ersetzen durch die Anzeige `Exchange?` zur Bestätigung des Vorgangs auf. Geben Sie Y ein, um den Austausch vorzunehmen oder brechen Sie mit einer anderen Taste ab.

Global Query-Replace...	Esc,R,P, E,Q...	Sucht nach jedem Vorkommen des auszutauschenden Textes und fragt jeweils nach einer Bestätigung. Geben Sie Y ein, um den Austausch vorzunehmen, oder brechen Sie mit einer anderen Taste ab.

4.1.3.6 Menü "Settings"

Mit den folgenden Befehlen wird die ED-Umgebung eingestellt:

Set FN Key...	Esc,S,F...	Belegt die Funktionstasten und andere programmierbare Tasten. Das Definieren von Befehlen für Funktionstasten und Tastenkombinationen mit Ctrl ähnelt dem Definieren von Menüpunkten. Kap. 4.1.3.7 enthält Anweisungen zur Belegung von Funktionstasten und ein Beispiel für einen "Set FN Key "-Befehl.
Show FN Key... (Display FN)	Esc,D,F <Taste>	Zeigt die Belegung der durch <Taste> angegebenen Funktionstaste an. Geben Sie für <Taste> ein Leerzeichen und eine Platzhalternummer an.
Reset Keys	Esc,R,K	Setzt die Tastendefinitionen auf die Standardwerte zurück. Kap. 4.1.3.8 zeigt eine Tabelle mit den Belegungen der Sondertasten.
Right Margin... (Set Right)	Esc,S, R...	Setzt den rechten Rand. Verwenden Sie den SR-Befehl mit einer Zahl, die die Spaltenposition angibt.
Left Margin... (Set Left)	Esc,S,L...	Setzt den linken Rand. Verwenden Sie den SL-Befehl mit einer Zahl, die die Spaltenposition angibt. Der linke Rand darf nicht rechts vom rechten Bildschirmrand gesetzt werden.
Ignore Case	Esc,U,C	Hiermit wird die Suche unabhängig von der Schreibweise durchgeführt. Mit UC wird bei allen nachfolgenden Suchbefehlen nicht zwischen Groß- und Kleinschreibung unterschieden. Soll die Suche wieder zwischen Groß- und Kleinschreibung unterscheiden, geben Sie den Befehl LC ein.

Case Sensitive Esc,L,C Hiermit wird die Suche abhängig von der Schreibweise durchgeführt.

4.1.3.7 Befehl "Set FN Key"

Mit dem Befehl "Set FN Key" werden Funktionstasten und andere programmierbare Tasten belegt. 57 Platzhalter für Befehlstasten von 1 bis 57 stehen zur Verfügung. Jede Platzhalternummer kann neu belegt werden, und alle Nummern innerhalb dieses Bereichs, die nicht in der Tabelle der Belegungen der Sondertasten in Kap. 4.1.3.8 aufgeführt sind, sind noch nicht definiert.

Für den Befehl "Set FN Key" gilt folgende Syntax:

```
SF <Platzhalternummer> /Befehl/
```

Verwenden Sie zur Definition einer Ctrl-Tastenkombination ein Winkelzeichen, auch Caret genannt, (^) und den jeweiligen Buchstaben anstelle der Platzhalternummer.

Beispiel einer Befehlsdatei

Dieses Beispiel-Skript ordnet einigen Funktionstasten zur Cursorsteuerung bestimmte Befehle zu. Dies kann auch als eine Reihe von Befehlen im erweiterten Modus eingegeben werden. Den Tasten F1 bis F6 werden die Befehle Top of File (Dateianfang), Bottom of File (Dateiende), End of Page (Seitenende), Next Page (Nächste Seite), Next Line (Nächste Zeile) und Previous Line (Vorhergehende Zeile) zugeordnet. Als Begrenzungszeichen werden Anführungszeichen verwendet.

```
SF 1"t"
SF 2"b"
SF 3"ep"
SF 4"pd"
SF 5"n"
SF 6"p"
```

4.1.3.8 Belegungen der Sondertasten

Die folgende Tabelle zeigt die Standardtastenbelegungen, die im Befehl "Reset Keys"verwendet werden:

Platzhalter	Taste/Tastenfolge	Funktion
1-10	F1 bis F10	Nicht belegt
11-20	Shift-F1 bis Shift-F10	Nicht belegt
21	Shift-Cursor links	Zeilenanfang
22	Shift-Cursor rechts	Zeilenende
23	Shift-Cursor aufwärts	Dateianfang
24	Shift-Cursor abwärts	Dateiende
25	Del	Aktuelles Zeichen löschen
26	Nicht definiert	Nicht belegt
27	Ctrl-A	Zeile einfügen
28	Ctrl-B	Zeile löschen
29	Ctrl-C	Nicht belegt
30	Ctrl-D	12 Zeilen nach unten
31	Ctrl-E	Oberer oder unterer Bildschirmrand
32	Ctrl-F	Groß- und Kleinschreibung vertauschen
33	Ctrl-G	Letzten erweiterten Befehl wiederholen
34	Ctrl-H	Zeichen links vom Cursor löschen
35	Ctrl-I	Nächster Tabulator
36	Ctrl-J	Nicht belegt
37	Ctrl-K	Nicht belegt
38	Ctrl-L	Nicht belegt
39	Ctrl-M	Neue Zeile
40	Ctrl-N	Nicht belegt
41	Ctrl-O	Wort oder Leerzeichen löschen
42	Ctrl-P	Nicht belegt
43	Ctrl-Q	Nicht belegt
44	Ctrl-R	Ende des vorhergehenden Worts
45	Ctrl-S	Nicht belegt
46	Ctrl-T	Anfang des nächsten Worts
47	Ctrl-U	12 Zeilen nach oben
48	Ctrl-V	Fenster erneut anzeigen

Platz-halter (Forts.)	Taste/Tastenfolge (Forts.)	Funktion (Forts.)
49	Ctrl-W	Nicht belegt
50	Ctrl-X	Nicht belegt
51	Ctrl-Y	Bis Zeilenende löschen
52	Ctrl-Z	Nicht belegt
53	Ctrl-[	Esc (Eingabe erweiterter Befehle)
54	Nicht definiert	Nicht belegt
55	Ctrl-]	Zeilenanfang oder -ende (je nach Cursorposition)
56	Nicht definiert	Nicht belegt
57	Nicht definiert	Nicht belegt

4.1.3.9 Menü "Command"

Die folgenden Befehle haben mit Befehlsdateien zu tun.

Extended Command...	Esc,C,M...	Schaltet in den Modus für erweiterte Befehle; entspricht Ctrl-[oder Esc.
Repeat Last	Esc,R,E	Der letzte Befehl soll wiederholt werden.
Run File...	Esc,R,F...	Lädt und führt eine Befehlsdatei mit erweiterten Befehlen aus.
ARexx Command...	Esc,R,X...	Führt das angegebene ARexx-Programm aus.
Redisplay	Esc,V,W	Baut das ED-Fenster neu auf und löscht den Inhalt der Statuszeile. Entspricht Ctrl-V.

4.1.3.10 Sonstige ED-Befehle

Neben den in den Menüs gezeigten Befehlen verfügt ED noch über weitere Befehle. Diese Befehle werden im folgenden sortiert nach Funktionsgruppen aufgeführt. Sie werden im erweiterten Modus durch Eingabe der angegebenen Tastenfolgen aufgerufen.

Programmsteuerung

Die Befehle zur Programmsteuerung sind:

Extend Margins	Esc,E,X	Erweitert die Ränder der aktuellen Zeile. Nach Eingabe des EX-Befehls wird der rechte Rand der aktuellen Zeile ignoriert.
Status Line Message	Esc,S,M...	Eine vorgegebene Zeichenfolge wird in der Statuszeile ausgegeben.
Exit with Query	Esc,X,Q	Verläßt ED, wenn keine Änderungen an der Datei vorgenommen wurden. Wenn etwas geändert wurde, wird die Meldung **`File has been changed-type Y to save and exit:`** (Datei wurde geändert - Mit Y Datei speichern und verlassen) angezeigt. Drücken Sie eine beliebige Taste (außer Y), um den Vorgang abzubrechen. Der Befehl XQ hat die gleiche Funktion wie das Klicken auf das Schließsymbol im ED-Fenster.

Plazieren des Cursors

Mit folgenden Befehlen kann der Cursor plaziert werden:

End Page	Esc,E,P	Seitenende.
Previous	Esc,P	Anfang der vorhergehenden Zeile.
Character Left	Esc,C,L	Eine Stelle nach links.
Character Right	Esc,C,R	Eine Stelle nach rechts.
Current End	Esc,C,E	Zeilenende.
Current Start	Esc,C,S	Zeilenanfang.
Tab	Esc,T,B	Nächster Tabulator.
Word Next	Esc,W,N	Anfang des nächsten Worts.
Word Previous	Esc,W,P	Leerzeichen nach vorhergehendem Wort.

Textbearbeitung

Mit den folgenden Befehlen können Sie Text auf dem Bildschirm edieren:

Insert Before	Esc,I...	Die angegebene Zeichenfolge wird in der Zeile über dem Cursor eingefügt. Geben Sie nach dem I-Befehl die Zeichenfolge an, die als neue Zeile vor der aktuellen eingefügt werden soll.
Insert After	Esc,A...	Die angegebene Zeichenfolge wird in der Zeile unter dem Cursor eingefügt. Dieser Befehl entspricht dem I-Befehl, die Zeichenfolge wird hier jedoch als neue Zeile unter dem Cursor eingefügt.
Split	Esc,S	Trennt die aktuelle Zeile an der Cursorposition.
Join	Esc,J	Verknüpft die nächste mit der aktuellen Zeile.
Delete	Esc,D	Löscht die aktuelle Zeile.
Delete Character	Esc,D,C	Löscht das Zeichen unter dem Cursor.
Delete Left	Esc,D,L	Löscht das Zeichen links des Cursors.
Delete Word	Esc,D,W	Löscht bis zum Ende des aktuellen Worts.
End Line	Esc,E,L	Löscht bis zum Ende der aktuellen Zeile.
Flip Case	Esc,F,C	Schaltet jeweils buchstabenweise in Groß- bzw. Kleinschreibung um.
Set Tab	Esc,S,T	Setzt den Tabulator. Soll der aktuelle Tabulator geändert werden, hinter dem ST-Befehl eine Zahl eingeben.
Next	Esc,N	Anfang der nächsten Zeile.

4.1.4 Befehlswiederholung im erweiterten Modus

Wenn Sie Ctrl-G drücken, wird eine Befehlszeile wiederholt. Hiermit können Sie komplexe Folgen von Edierbefehlen definieren und mehrere Male ausführen.

Sie können auch festlegen, wie oft Sie einen erweiterten Befehl wiederholen wollen, indem Sie vor dem Befehl eine Zahl eingeben. Zum Beispiel wird mit

```
4 E/umbenennen/kopieren/
```

das Wort "umbenennen" viermal durch "kopieren" ersetzt.

Sie können mit dem erweiterten Befehl RP (Repeat) einen Befehl so lange wiederholen, bis ED einen Fehler meldet, z. B. wenn das Dateiende erreicht ist. Zum Beispiel wird mit

```
T;RP E/umbenennen/kopieren/
```

der Cursor an den Dateianfang gestellt und "umbenennen" generell durch "kopieren" ersetzt. Beachten Sie, daß Sie den T-Befehl benötigen, wenn Sie das Wort "umbenennen" in der gesamten Datei austauschen wollen. Andernfalls wird es nur ab der aktuellen Cursorposition ersetzt.

Wenn Sie wiederholt mit ganzen Befehlsgruppen arbeiten wollen, können Sie die Befehle mit Klammern zusammenfassen. Befehle können auch verschachtelt werden. Zum Beispiel werden mit

```
RP (F/Workbench/;3A//)
```

hinter jeder Zeile, die das Wort "Workbench" enthält, drei Leerzeilen (durch //) eingefügt.

Wenn Sie eine Folge erweiterter Befehle unterbrechen wollen, drücken Sie während der Ausführung der Befehle eine beliebige Taste. Wenn ein Fehler auftritt, verläßt ED die Befehlsfolge.

4.1.5 Anpassen von ED

In diesem Abschnitt werden die Befehle zur Konfiguration der Menüs und Funktionstasten beschrieben. Diese Befehle können in ED unabhängig eingegeben werden, sie können aber auch als Befehlsdatei gespeichert werden, z. B. als "S:Ed-startup" oder als Datei, die mit dem Argument WITH angegeben wird. Um diese Datei aufzurufen, verwenden Sie den erweiterten Befehl Esc,R,F (Starten einer Datei). Näheres über das Umbelegen der Funktionstasten siehe Kap. 4.1.3.7.

Set Menu Item	Esc,S,I	Definiert die Titel und Menüpunkte der Menüs. Es gibt insgesamt 120 Platzhalter für Menüpunkte (durchnumeriert von 0 bis 119). Der Platzhaltertyp kennzeichnet den Inhalt des Platzhalters und ist eine Ziffer von 0 bis 4. Platzhaltertyp 0 muß der letzte definierte Platzhalter sein. Achten Sie darauf, kein Menü zu erstellen, das keine Menüpunkte enthält. Wenn Sie einen Menütitel angeben, müssen darauf auch Menüpunkte folgen. Weiter unten finden Sie die Syntax des Befehls "Set Menu Item" sowie eine Tabelle mit einer Beschreibung der Platzhaltertypen.
Enable Menu	Esc,E,M	Aktiviert Menüs. Auf die Befehle von "Set Menu Item" muß EM folgen, damit die Menübefehle aktiviert werden. Auf Seite 4-25 sehen Sie ein Beispielprogramm, in dem der Befehl "Enable Menu" verwendet wird.

4.1.5.1 Set Menu Item (Menüpunkt setzen)

Folgendes ist die Syntax des Befehls "Set Menu Item":

```
SI <Nummer des Platzhalters> <Platzhaltertyp>
/Zeichenfolge1/ /Zeichenfolge2/
```

Typ	Funktion	Zeichenfolge
0	Menüende	Keine Argumente
1	Menütitel	Zeichenfolge1 = Titel
2	Menüpunkt	Zeichenfolge1 = Text des Menüpunkts Zeichenfolge2 = Befehl
3	Untermenütitel	Zeichenfolge1 = Titel Zeichenfolge2 = Befehl
4	Trennbalken	Keine Argumente

Beispiel einer Befehlsdatei

Es folgt ein Beispiel für eine Befehlsdatei, in der die Befehle "Set Menu Item" und "Enable Menu" verwendet werden. Als Begrenzungszeichen werden Anführungszeichen verwendet.

```
SI  0 1 "Project"
SI  1 2 "Open ... " "op ? /Open file:/"
SI  2 2 "Save ... " "sa"
SI  3 4
SI  4 2 "Quit!" "q"
SI  5 1 "Move"
SI  6 2 "Top" "t"
SI  7 2 "Bottom" "b"
SI  8 0
EM
```

Mit dieser Datei werden die in Abbildung 4-4 dargestellten Menüs erstellt.

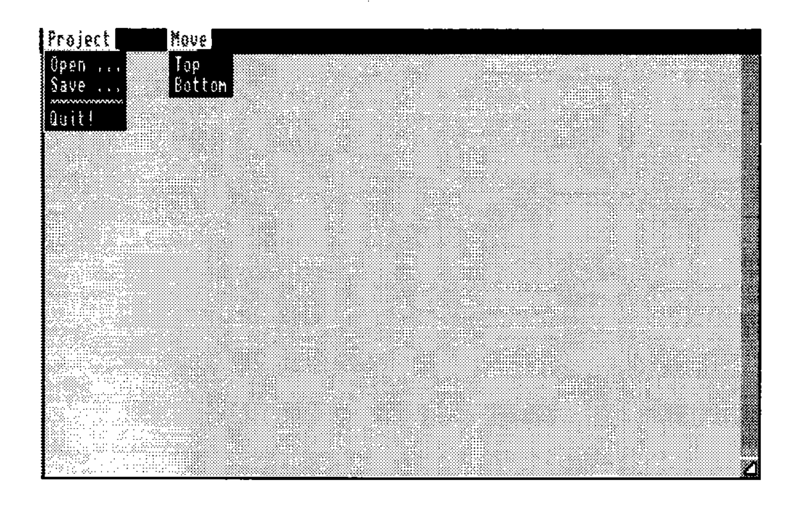


Abbildung 4-4. Angepaßtes ED-Menü - Beispiel

4.1.6 Drucken aus ED

Mit folgenden Schritten können Sie eine Datei drucken, die im aktuellen ED-Fenster geladen ist:

1. Wählen Sie den Menüpunkt "Save As" zur Anzeige eines Dateiauswahlfensters aus.
2. Geben Sie **prt:** in das Schubladenfeld (Drawer) ein.
3. Wählen Sie OK aus.

Hiermit wird die Datei gedruckt, aber nicht auf Disk gespeichert. Soll die Datei gespeichert werden, muß der Menüpunkt "Save" oder "Save As" und der Dateiname ausgewählt werden.

4.1.7 Verlassen von ED

Sie können ED auf eine der drei folgenden Arten verlassen:

- Esc,X. Bei dieser Methode wird ED verlassen und die aktuelle Datei unter einem bestimmten Dateinamen, der bei Aufruf von ED mit angegeben wurde, gespeichert.
- Esc,X,Q oder klicken auf das Schließsymbol im ED-Fenster. Bei dieser Methode wird ED verlassen, wenn keine Änderungen an der Datei vorgenommen wurden. Wenn etwas geändert wurde, können Sie die Änderungen speichern oder ED ohne Speichern verlassen.
- Esc,Q oder Auswählen des Menüpunkts "Quit" im Menü "Project". Bei dieser Methode wird ED ohne Speichern der Änderungen verlassen, wenn Sie die Warnung in der Statuszeile, die besagt, daß alle Änderungen verloren gehen, bestätigen.

4.1.8 ARexx-Unterstützung

ED kann auch über ARexx gesteuert werden, indem Befehle über den ARexx-Port von ED gesendet und empfangen werden. Jede laufende ED-Sitzung hat für ihren ARexx-Port einen eigenen Namen, der angegeben werden muß, damit Informationen für die jeweilige Sitzung bearbeitet werden können. Die Namen für die ARexx-Ports werden wie folgt vergeben:

- Der Port-Name für die erste Sitzung lautet Ed
- Für die zweite Sitzung Ed_1
- Für die dritte Sitzung Ed_2 und so weiter

  Viele der erweiterten ED-Befehle können Sie auch von ARexx aus geben. Mit dem ED-Befehl RV in ARexx-Programmen können Informationen von ED an ARexx übertragen werden. Sie können so den Status von ED abrufen (z. B. die Nummer der aktuellen Zeile oder den Namen der Datei, die gerade ediert wird.)

Bei dem Befehl RV wird als Argument der Name der ARexx-Variablen "stem" angegeben, in der die Informationen gespeichert werden sollen. Zum Beispiel in ARexx werden mit

```
address 'Ed' 'RV /stem/'
```

den folgenden Variablen Werte zugeordnet:

stem.LEFT	Linker Rand
stem.RIGHT	Rechter Rand
stem.TABSTOP	Tabulatoreinstellung
stem.LMAX	Maximale Anzahl sichtbarer Zeilen auf dem Bildschirm
stem.WIDTH	Bildschirmbreite in Zeichen
stem.X	X-Position des Cursors im ED-Fenster (1 entspricht dem linken Rand)
stem.Y	Y-Position des Cursors im ED-Fenster (1 entspricht der ersten Zeile)
stem.BASE	Fensterbasis (normalerweise 0, wenn das Fenster nicht nach rechts verschoben wurde)
stem.EXTEND	Wert des erweiterten Rands (Befehl "Extend Margins")
stem.FORCECASE	Kennung für Groß- und Kleinschreibung (1 = ignorieren, 0 = unterscheiden)
stem.LINE	Nummer der aktuellen Zeile (1 entspricht der ersten Zeile)
stem.FILENAME	Name der aktuellen Datei
stem.CURRENT	Inhalt der aktuellen Zeile
stem.LASTCMD	Zuletzt ausgeführter erweiterter Befehl
stem.SEARCH	Letzte gesuchte Zeichenfolge

Sie können für "stem" auch jedes andere gültige ARexx-Symbol verwenden. Geben Sie dieses aber zwischen Begrenzungszeichen ein. Diese Variablen können dann als normale "stem"-Variablen verwendet werden.

4.1.8.1 ED/ARexx-Beispielprogramm

Das Beispielprogramm "Transpose.ed" zeigt die Verwendung verschiedener erweiterter Befehle von ARexx. Dieses Programm stellt zwei Zeichen um, wenn es von ED aus aufgerufen wird. Enthält eine Zeile zum Beispiel die Zeichenfolge 123 und steht der Cursor auf der 3, ändert Transpose.ed die Zeichenfolge in 213.

Geben Sie dieses Programm ein und speichern Sie es unter REXX:Transpose.ed. Öffnen Sie dann ED und edieren Sie eine vorhandene Datei oder erstellen Sie eine neue. Stellen Sie den Cursor rechts von den beiden Zeichen, die umgestellt (transponiert) werden sollen, drücken Sie Esc und geben Sie folgendes ein:

```
RX /transpose.ed/
```

Das Programm wird ausgeführt, und die Zeichen werden umgestellt, wenn ARexx läuft und alle Angaben richtig eingegeben wurden. Zur Ausführung des Programms muß der gesamte Dateiname einschließlich Erweiterung eingegeben werden.

Beispielprogramm

```
/*Transpose.ed: Beispielprogramm zum Vertauschen zweier Zeichen          */
/* Gegeben: Zeichenfolge '123', Bedingung: Cursor steht auf 3;           */
/* Ergebnis: Zeichenfolge '213'.                                         */
HOST = address ()          /* Welche ED-Sitzung hat Programm aufgerufen? */
address VALUE HOST         /*.Mit dieser Sitzung kommunizieren           */
'rv' '/CURR/'              /* Daten in stem-Variable CURR speichern      */
                           /* Zwei Informationen abrufen:                */
currpos = CURR.X                  /* 1. Position des Cursors in Zeile    */
currlin = CURR.CURRENT            /* 2. Inhalt der aktuellen Zeile       */
if (currpos > 2) then             /* Nur in aktueller Zeile              */
  currpos = currpos - 1
else do                           /* Sonst: Fehlermeldung und verlassen  */
'sm /Cursor muß mindestens auf Position 2 stehen/'
exit 10
end
/* Die Zeichen auf CURRPOS und CURRPOS-1 werden dann vertauscht,         */
/* und die aktuelle Zeile wird durch die neue ersetzt..                  */
drop CURR /*.  CURR nicht mehr erforderlich; Löschen spart Speicher      */
'd'                                /* ED löscht aktuelle Zeile           */
currlin = swapch (currpos,currlin) /* Zeichen vertauschen                */
'i /'||currlin||'/'                /* Geänderte Zeile einfügen           */
do i = 1 to currpos                /* Cursor an Ausgangsposition setzen  */
  'cr'                             /* Befehl 'Cursor nach rechts'        */
  end
exit                               /* Programm beendet                   */
/* Funktion zum Vertauschen der Zeichen                                  */
swapch: procedure
parse arg cpos,clin
  ch1 = substr(clin,cpos,1)        /* Zeichen erfassen                   */
  clin = delstr(clin,cpos,1)       /* Aus Zeichenfolge löschen           */
```

```
  clin = insert(ch1,clin,cpos-2,1)      /* Wieder einfügen                   */
return clin                             /* Geänderte Zeichenfolge zurückgeben */
```

4.2 MEmacs

Bei MEmacs (MicroEmacs), der dem unter UNIX erkannten Emacs-Editor ähnelt, handelt es sich um einen Bildschirmeditor, mit dem Sie mehrere Dateien gleichzeitig bearbeiten können. Die einzige Einschränkung besteht darin, daß die gesamte Textdatei vollständig in den Speicher passen muß, da in MEmacs Operationen nur in speicherresidentem Text ausgeführt werden.

Die Länge der edierbaren Zeilen ist auf den rechten Rand des Bildschirms begrenzt. Dies entspricht normalerweise 80 Zeichen. Zeichen, die darüber hinaus eingegeben werden, werden zwar akzeptiert, sie erscheinen aber nicht mehr auf dem Bildschirm. Sie können sich diese Zeichen anzeigen lassen, indem Sie die Zeile umbrechen oder einige Zeichen löschen. Ein Dollarzeichen ($) am rechten Bildschirmrand bedeutet, daß noch weitere Zeichen folgen, die nicht mehr auf dem Bildschirm erscheinen.

Das Format des MEmacs-Befehls lautet:

MEMACS [<Dateiname>] [*GOTO* <n>] [*OPT* W]

Das Argument <Dateiname> muß nicht angegeben werden.

Die Option GOTO <n> gibt die Zeile an, in der der Cursor beim Öffnen der Datei stehen soll.

Die Angabe OPT W sorgt dafür, daß MEmacs nicht auf einem eigenen Schirm, sondern in einem Workbench-Fenster geöffnet wird; das spart Arbeitsspeicher.

4.2.1 Starten von MEmacs

Sie können MEmacs von der Workbench oder von der Shell aus starten. Von der Workbench aus klicken Sie doppelt auf das Piktogramm "MEmacs" im Fenster "Tools" der Disk "Extras". Wenn Sie mit einer Festplatte arbeiten, finden Sie die Schublade "Tools" im Workbench-Fenster.

Von der Shell aus geben Sie den Befehl im folgenden Format ein:

```
MEmacs <dateiname>
```

Mit <dateiname> geben Sie die Datei an, die in MEmacs geladen werden soll. Wenn keine Datei mit diesem Namen gefunden wird, wird beim Speichern eine neue erstellt.

4.2.2 MEmacs-Befehle

In der Zeile am unteren Rand der MEmacs-Anzeige steht entweder der aktuelle Dateiname oder, wenn kein Dateiname angegeben wurde, der Name des aktuellen Puffers. Abbildung 4-5 zeigt die MEmacs-Startanzeige.

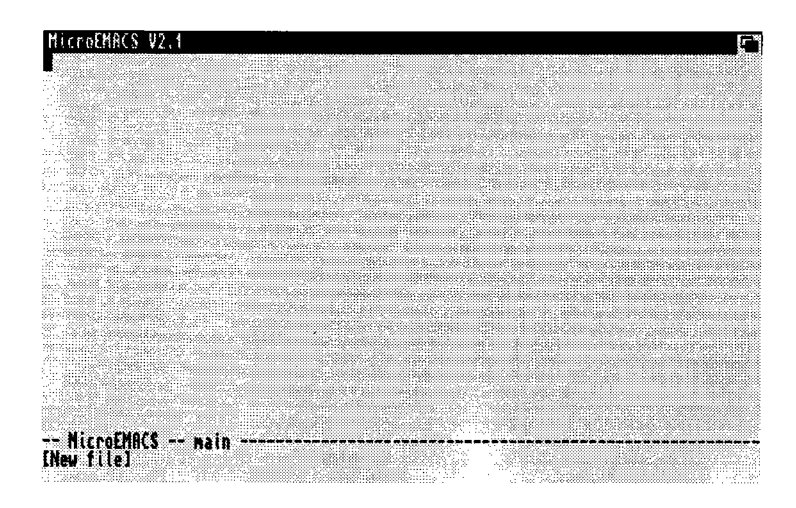


Abbildung 4-5. MEmacs-Startanzeige

Sie können gleichzeitig mit mehreren Pufferspeichern arbeiten und auf dem Bildschirm anzeigen lassen. Über die Menüpunkte können Sie zwischen ihnen hin- und herschalten. Dabei entspricht die aktuelle Bildschirmanzeige dem Inhalt des jeweiligen Puffers.

MEmacs verfügt über zwei Betriebsmodi:

Normalmodus Direkte Eingabe und Bearbeitung von Text in der Datei, ohne Sonderfunktionen.

Befehlsmodus Eingabe eines Befehls über eine Menüauswahl oder ein Tastaturkürzel. Im Befehlsmodus springt der Cursor in die unterste Zeile der Anzeige und wartet auf Benutzereingaben zusätzlicher Informationen hinter der Eingabeaufforderung. Erst nach Ausführung oder Abbruch des Befehls durch Drücken der Eingabetaste können Sie wieder in den Normalmodus zurückschalten.

Im Normalmodus von MEmacs können Sie:

- Mit den Cursortasten den Cursor plazieren.
- Mit Shift und der entsprechenden Cursortaste den Cursor an den Rand des Fensters stellen
- Durch Klicken der linken Maustaste den Cursor an die gewünschte Stelle auf dem Bildschirm setzen
- Zeichen an der aktuellen Cursorposition einfügen
- Mit der Del-Taste das Zeichen an der aktuellen Cursorposition löschen
- Mit der Rücktaste das Zeichen links vom Cursor löschen
- Weitere Funktionen ausführen, die in den folgenden Abschnitten näher erläutert werden.

Für das Arbeiten mit MEmacs sollten Sie mit den folgenden Begriffen vertraut sein:

Buffer Ein von MEmacs verwalteter Speicherbereich (Pufferspeicher). MEmacs verwendet immer mindestens einen Pufferspeicher mit null oder mehr Textzeichen.

Dot Aktuelle Cursorposition.

Mark Eine von Ihnen angegebene Cursorposition (Markierung). (Jeder Pufferspeicher hat sein eigenes "Dot" und "Mark".) Mit dem Menüpunkt "Set-mark" (Markierung setzen) markieren Sie die aktuelle Cursorposition (siehe Seite 4-37). Nun können Sie die Datei normal bearbeiten (Cursor bewegen, Text hinzufügen oder löschen). Wenn Sie später an diese Stelle zurückkehren wollen, wählen Sie den Menüpunkt "Swap-dot&mark" (Punkt und Markierung vertauschen) (siehe Seite 4-40).

Sie können damit auch den Anfang eines Textblocks kennzeichnen, den Sie kopieren, verschieben oder löschen wollen. Der Block umfaßt alle Zeichen von der Markierung bis zur aktuellen Cursorposition.

Kill Mit dem Befehl "Kill" löschen Sie Text vom Bildschirm und speichern ihn in einem Kill-Pufferspeicher. Mit dem Befehl "Yank" können Sie den Text wieder in das Dokument einfügen (siehe entsprechenden Abschnitt unten). Wenn Sie den Befehl "Kill" mehrmals hintereinander aufrufen (ohne dazwischen "Yank" zu verwenden), werden die einzelnen Textblöcke jeweils dem bestehenden Text im Kill-Pufferspeicher hinzugefügt. Nach Auswahl von "Yank" überschreibt der nächste in den Kill-Puffer gestellte Text den aktuellen Textblock.

Window In MEmacs kann der Bildschirm in mehrere horizontale Bereiche unterteilt werden. So können Sie mehr als einen Pufferspeicher oder auch zwei oder mehr Teilbereiche eines Pufferspeichers edieren und anzeigen lassen. Jeder Bereich ist ein MEmacs-Fenster.

Modified Buffers Wenn an einem Puffer Änderungen vorgenommen werden, wird dieser als "modified" (geändert) gekennzeichnet. Beim Speichern des Puffers wird diese Kennzeichnung wieder entfernt.

Mit dem Befehl "List-buffers" (siehe Seite 4-37) wird eine Liste der aktuellen Pufferspeicher angezeigt, der Sie entnehmen können, welche geändert wurden. Geänderte Pufferspeicher sind mit einem Sternchen (*) gekennzeichnet. Wenn Sie MEmacs verlassen wollen, ohne die Änderungen zu speichern, wird eine Meldung mit einem entsprechenden Hinweis angezeigt. Geben Sie ein, ob Sie das Programm dennoch verlassen wollen.

4.2.3 Menübefehle

In MEmacs sind folgende Menüs verfügbar:

Project Enthält system- und dateibezogene Menüpunkte.

Edit Enthält Befehle zum Edieren der Pufferspeicher.

Window Steuert die Anzeige von MEmacs-Fenstern.

Move Cursorbewegungen.

Line Zeilenbezogene Operationen.

Word Wortbezogene Operationen.

Search Such- und Ersetzungsoperationen.

Extras Bestimmt den numerischen Wert eines Arguments und ermöglicht die Ausführung mehrerer Funktionen in einem Arbeitsgang.

Abbildung 4-6 zeigt die erweiterte Menüleiste von MEmacs.

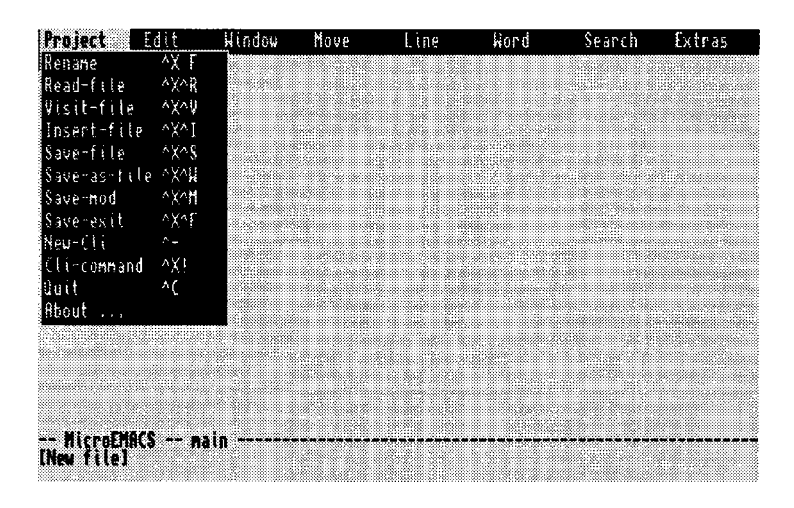


Abbildung 4-6. Erweiterte Menüs in MEmacs

4.2.3.1 Menü "Project"

Mit Ausnahme des Menüpunkts "Visit-file" beziehen sich alle Befehle im Menü "Project" auf den Pufferspeicher mit dem aktiven Cursor.

Rename Ctrl-X,F Der Name der Datei, die dem aktuellen Pufferspeicher zugeordnet ist, wird geändert. Wenn Sie die Eingabetaste drücken, ohne einen Dateinamen anzugeben, wird die Zuordnung zwischen Pufferspeicher und Dateiname aufgehoben.

Read-file	Ctrl-X, Ctrl-R	Der Inhalt des aktuellen Pufferspeichers wird durch den Inhalt einer Datei ersetzt. Geben Sie den vollständigen Pfad der Datei ein. Wenn Sie keine Datei einlesen wollen, drücken Sie die Eingabetaste, ohne einen Dateinamen einzugeben. Der Vorgang wird dann abgebrochen, und Sie kehren in den normalen Modus zurück.
Visit-file	Ctrl-X, Ctrl-V	Sie können außer mit der ersten geöffneten Datei auch noch mit zusätzlichen Dateien arbeiten. Geben Sie den vollständigen Pfad ein.
Insert-file	Ctrl-X, Ctrl-I	Der Inhalt einer Datei wird in den aktuellen Pufferspeicher eingefügt, und zwar in der Zeile über der aktuellen Cursorposition. Geben Sie den vollständigen Pfad ein.
Save-file	Ctrl-X, Ctrl-S	Der Inhalt des aktuellen Pufferspeichers wird in die Datei übertragen, die dem Puffer zugeordnet ist. Nach erfolgter Speicherung wird angezeigt, wie viele Zeilen in die Datei übertragen wurden. MEmacs speichert die Datei nicht, wenn kein Dateiname angegeben wird. In diesem Fall erscheint die Fehlermeldung: `No file name` (Kein Dateiname).
Save-as-file	Ctrl-X, Ctrl-W	Sie können eine Datei (Name und Pfad) angeben, die Sie einem Pufferspeicher zuordnen wollen.
Save-mod	Ctrl-X, Ctrl-M	Der Inhalt aller geänderter Pufferspeicher wird auf Disk gespeichert. Verwenden Sie diese Funktion nur, wenn Sie damit keinen versehentlich geänderten Pufferspeicher mitspeichern.
Save-exit	Ctrl-X, Ctrl-F	Alle geänderten Pufferspeicher werden gespeichert, und MEmacs wird verlassen.
New-Cli	Ctrl--	Es erscheint ein neues Shell-Fenster mit dem Namen "Spawn Window". In diesem Fenster können Sie unabhängig von MEmacs AmigaDOS-Befehle verwenden. Mit dem Befehl ENDSHELL können Sie zu MEmacs zurückkehren.

Cli-Command	Ctrl-X,!	Sie können innerhalb von MEmacs AmigaDOS-Befehle aufrufen. Geben Sie den Befehl hinter der Eingabeaufforderung ! (Ausrufezeichen) unten in der Anzeige ein. Die Ausgabe des Befehls wird im temporären Pufferspeicher "spawn.output" abgelegt.
Quit	Ctrl-C	MEmacs wird verlassen. Sie erhalten die Möglichkeit, geänderte Pufferspeicher vorher zu speichern oder ohne Speichern der Änderungen das Programm zu verlassen. Alternative Tastaturkürzel hierfür sind: Ctrl-X,Ctrl-C Esc,Ctrl-C.
About...		Zeigt urheberrechtliche Informationen zu dem Programm an.

4.2.3.2 Menü "Edit"

Die Befehle im Menü "Edit" verwenden Sie, wenn Sie Pufferspeicher und die zugeordneten Dateien edieren wollen.

Kill-region	Ctrl-W	Aus dem aktuellen Pufferspeicher werden Textblöcke gelöscht und in einem Kill-Pufferspeicher gespeichert. Mit dem Befehl "Yank" können Sie diesen Text wieder in das Dokument einfügen. Sie können mit dem Befehl "Kill-region" auch einen Block innerhalb des Pufferspeichers kopieren. Markieren Sie den Block, wählen Sie "Kill-region" und dann "Yank" aus, ohne die Position des Cursors zu ändern. Der Block wird an seiner ursprünglichen Position gleich wieder eingefügt. Es verbleibt jedoch eine Kopie des Blocks im Kill-Pufferspeicher.

Yank	Ctrl-Y	Der Inhalt des Kill-Pufferspeichers wird in die Zeile mit der aktuellen Cursorposition im Pufferspeicher kopiert. Der Befehl "Yank" kehrt die Aktion des Befehls "Kill-region" um, jedoch wird der Inhalt des Kill-Pufferspeichers nicht geändert. Der Befehl dient in Verbindung mit "Kill-region" dazu, Text zu verschieben oder einen Textblock mehrmals zu kopieren.
Set-mark	Ctrl-@	Die Cursorposition in einem Pufferspeicher wird markiert. Jede andere Position, an der sich der Cursor dann befindet, wird als "dot" (Punkt) bezeichnet. Sie können durch den Befehl "Swap-dot&mark" im Menü "Move" mit dem Cursor zwischen Markierung und "dot" hin- und herspringen. Der Befehl wird zum Markieren von Textblöcken verwendet. Entspricht dem Tastaturkurzbefehl "Esc,-".
Copy-region	Esc,W	Der markierte Bereich wird in den Kill-Pufferspeicher kopiert, ohne gelöscht zu werden. Hierbei wird der bisherige Inhalt des Pufferspeichers überschrieben.
Upper-region	Ctrl-X, Ctrl-U	Der gesamte markierte Text (zwischen Markierung und aktueller Cursorposition) wird in Großbuchstaben umgesetzt.
Lower-region	Ctrl-X, Ctrl-L	Der gesamte markierte Text wird in Kleinbuchstaben umgesetzt.
List-buffers	Ctrl-X, Ctrl-B	Das Fenster des aktuellen Pufferspeichers wird geteilt und zeigt eine Liste der in MEmacs verfügbaren Pufferspeicher an. Wählen Sie den Befehl "One window" aus oder drücken Sie Ctrl-X,1, um wieder den aktuellen Pufferspeicher anzuzeigen. Die Felder der Pufferspeicherliste haben folgende Bedeutung:

C	Ein Sternchen wird angezeigt, wenn der Pufferspeicher seit der letzten Speicherung geändert wurde (steht für "Changed").
Size	Anzahl der Zeichen im Pufferspeicher (Größe).

		Buffer Name des Pufferspeichers. Wenn eine Datei eingelesen wurde, ist dies der Name der Datei ohne Pfad.
		File Zeigt den ganzen Pfad zur Datei an. Dies ist die Datei, in der MEmacs den Pufferspeicher ablegt, wenn Sie "Save-file" oder "Save-exit" auswählen, während der Cursor sich noch in diesem Pufferspeicher befindet.
Select-buffer	Ctrl-X,B	Mit diesem Befehl wählen Sie den Pufferspeicher aus, den Sie im aktuellen Fenster edieren wollen. Hierbei wird entweder der Inhalt des Fensters mit dem angegebenen Pufferspeicher überschrieben oder ein neuer Pufferspeicher erstellt.
Insert-buffer	Esc, Ctrl-Y	Der Inhalt des angegebenen Pufferspeichers wird in den aktuellen Pufferspeicher in der Zeile über dem Cursor eingefügt.
Kill-buffer	Ctrl-X,K	Der Inhalt des ausgewählten Pufferspeichers wird gelöscht, wobei der Speicher wieder dem Speicherverwaltungsprogramm zur Verfügung gestellt wird. Der Name des zu löschenden Speichers muß dabei eingegeben werden. Ein Pufferspeicher kann nicht gelöscht werden, wenn er gerade angezeigt wird.
Justify-buffer	Ctrl-X,J	Alle Leerzeichen und Tabulatoren werden vom linken Rand aller Zeilen im aktuellen Pufferspeicher gelöscht. Der Text wird mit den aktuellen Randeinstellungen neu umgebrochen.
Redisplay	Ctrl-L	Der Bildschirm wird neu aufgebaut.
Quote-char	Ctrl-Q	Hiermit können Sie jedes Sonderzeichen (mit Ausnahme der Tabulatortaste) in die Textdatei einfügen. Entspricht dem Tastaturkurzbefehl Ctrl-X,Q.

Indent	Ctrl-J	Der Cursor wird in die nächste Zeile gestellt und rückt dabei ebenso viele Stellen ein wie in der Zeile darüber. Entspricht den Tastaturkurzbefehlen "Help" oder "Enter" auf dem Ziffernblock.
Transpose	Ctrl-T	Zwei nebeneinanderliegende Zeichen werden vertauscht. Stellen Sie den Cursor auf das rechte Zeichen und rufen Sie den Befehl auf.
Cancel	Ctrl-G	Der Menübefehl, der gerade ausgeführt wird (zum Beispiel Suchen und Ersetzen mit Rückfrage) wird abgebrochen.

4.2.3.3 Menü "Window"

Über das Menü "Window" legen Sie fest, wie die Pufferspeicher auf dem Bildschirm angezeigt werden.

One-window	Ctrl-X,1	Der aktuelle Pufferspeicher wird als ein einzelnes Fenster auf dem ganzen MEmacs-Bildschirm angezeigt.
Split-window	Ctrl-X,2	Das aktuelle Fenster wird in der Mitte geteilt. Der aktuelle Pufferspeicher wird in beiden Fenstern angezeigt. Änderungen in einem Fenster werden auch im anderen Fenster vorgenommen.
Next-window	Ctrl-X,N	Der Cursor wird in das nächste Fenster gestellt, in dem Sie dann edieren können.
Prev-window	Ctrl-X,P	Der Cursor wird in das vorhergehende Fenster gestellt, in dem Sie dann edieren können.
Expand-window	Ctrl-X,Z	Das aktuelle Fenster wird um eine Zeile vergrößert, während das benachbarte um eine Zeile verkleinert wird.
Shrink-window	Ctrl-X, Ctrl-Z	Das aktuelle Fenster wird um eine Zeile verkleinert, während das benachbarte um eine Zeile vergrößert wird.
Next-w-page	Esc, Ctrl-V	Die nächste Seite des benachbarten Fensters wird angezeigt. Ein Edieren in diesem Fenster wird hierdurch nicht möglich.

Prev-w-page	Ctrl-X,V	Die vorhergehende Seite des benachbarten Fensters wird angezeigt. Wenn nur ein Fenster angezeigt wird, wird zur vorhergehenden Seite dieses Fensters geblättert.

4.2.3.4 Menü "Move"

Mit den Befehlen im Menü "Move" können Sie den Cursor schnell im aktuellen Pufferspeicher plazieren.

Top-of-buffer	Esc,<	Der Cursor wird an den oberen Rand des aktuellen Pufferspeichers gestellt.
End-of-buffer	Esc,>	Der Cursor wird an den unteren Rand des aktuellen Pufferspeichers gestellt.
Top-of-window	Esc,,	Der Cursor wird an den oberen Rand des aktuellen Fensters gestellt.
End-of-window	Esc,.	Der Cursor wird an den unteren Rand des aktuellen Fensters gestellt.
Goto-line	Ctrl-X, Ctrl-G	Der Cursor wird in die angegebene Zeile gestellt. Wenn Sie eine Zahl angeben, die über der Anzahl der Zeilen im Pufferspeicher liegt, wird der Cursor in die letzte Zeile gesetzt.
Swap-dot&mark	Ctrl-X, Ctrl-X	Die aktuelle Cursorposition wird markiert und der Cursor an die zuvor markierte Stelle gestellt.
Next-page	Ctrl-V	Blättert im Pufferspeicher um ein Fenster minus eine Zeile weiter.
Prev-page	Esc,V	Blättert im Pufferspeicher um ein Fenster minus eine Zeile zurück.
Next-word	Esc,F	Der Cursor wird auf das nächste nicht alphanumerische Zeichen (z. B. Leerzeichen oder Satzzeichen) hinter dem aktuellen Wort gesetzt.
Previous-word	Esc,B	Der Cursor wird auf den ersten Buchstaben des vorhergehenden Worts gesetzt.
Scroll-up	Ctrl-Z	Der Text wird um eine Zeile nach oben gerollt.

Scroll-down	Esc,Z	Der Text wird um eine Zeile nach unten gerollt.

4.2.3.5 Menü "Line"

Mit den Befehlen im Menü "Line" können Sie den Cursor innerhalb von Zeilen oder zwischen Zeilen plazieren und mit ganzen Zeilen arbeiten.

Open-line	Ctrl-O	Die Zeile wird an der aktuellen Cursorposition geteilt. Das Zeichen, auf dem der Cursor steht, wird damit das erste Zeichen der neuen Zeile. Der Cursor bleibt in der ursprünglichen Zeile. Dieser Vorgang kann mit der Taste "Del" rückgängig gemacht werden.
Kill-line	Ctrl-X, Ctrl-D	Die Zeile, in der sich der Cursor befindet, wird gelöscht und in den Kill-Pufferspeicher aufgenommen.
Kill-to-eol	Ctrl-K	Der Text wird ab der aktuellen Cursorposition bis zum Zeilenende gelöscht und in den Kill-Pufferspeicher aufgenommen.
Start-of-line	Ctrl-A	Der Cursor wird an den Zeilenanfang gesetzt.
End-of-line	Ctrl-E	Der Cursor wird an das Zeilenende gesetzt.
Next-line	Ctrl-N	Der Cursor wird in die nächste Zeile gesetzt.
Previous-line	Ctrl-P	Der Cursor wird in die vorhergehende Zeile gesetzt.
Line-to-top	Esc,!	Die Zeile, in der sich der Cursor befindet, wird zur ersten Zeile des Fensters.
Delete-blanks	Ctrl-X, Ctrl-O	Alle Leerzeilen werden ab der aktuellen Cursorposition bis zur nächsten Zeile, die Text enthält, gelöscht. MEmacs löscht keine einzelnen Leerzeilen.
Show-Line#	Ctrl-X,=	Informationen zur aktuellen Cursorposition werden angezeigt.

4.2.3.6 Menü "Word"

Das Menü "Word" enthält wortbezogene Funktionen.

Delete-forw	Esc,D	Das Zeichen, auf dem sich der Cusor gerade befindet, und alle Zeichen rechts davon werden bis zum nächsten nicht alphanumerischen Zeichen (z. B. ein Leerzeichen, ein Tabulatorzeichen oder ein Interpunktionszeichen) gelöscht.
Delete-back	Esc,H	Alle Zeichen links vom Cursor bis zum ersten Zeichen des Worts werden gelöscht. Das Zeichen, auf dem der Cursor steht, wird nicht gelöscht. Entspricht dem Tastaturkurzbefehl Esc,Del.
Upper-word	Esc,U	Die Buchstaben des Worts werden ab der Cursorposition bis zum Wortende in Großbuchstaben umgesetzt.
Lower-word	Esc,L	Die Buchstaben des Worts werden ab der Cursorposition bis zum Wortende in Kleinbuchstaben umgesetzt.
Cap-word	Esc,C	Der Buchstabe, auf dem sich der Cusor befindet, wird in Großschreibung umgesetzt. Außerdem werden die Zeichen rechts vom Cursor bis zum Wortende in Kleinschreibung umgesetzt.
Switch-case	Esc,^	Die Schreibweise eines Worts wird ab der Cursorposition bis zum Wortende in Groß- bzw. Kleinschreibung umgesetzt. Großbuchstaben werden in Kleinschreibung umgesetzt und umgekehrt.

4.2.3.7 Menü "Search"

Mit dem Menü "Search" können Sie den aktuellen Pufferspeicher nach bestimmten Zeichenfolgen durchsuchen. Bei der Suche selbst spielt es keine Rolle, ob das Wort groß oder klein geschrieben ist. Wenn Sie jedoch mit Textersetzung arbeiten (Search and replace), wird der neue Text ebenso groß bzw. klein geschrieben wie die Zeichenfolge, die ersetzt wird.

Search-forward	Ctrl-S	Der Text wird ab der aktuellen Cursorposition bis zum Ende des Pufferspeichers durchsucht. Geben Sie nach der Eingabeaufforderung die Suchzeichenfolge ein. Entspricht dem Tastaturkurzbefehl Ctrl-X,S.
Search-backward	Ctrl-R	Der Text wird ab der aktuellen Cursorposition bis zum Anfang des Pufferspeichers rückwärts durchsucht. Entspricht dem Tastaturkurzbefehl Ctrl-X,R.
Search-replace	Esc,R	Entspricht in der Funktion dem Befehl "Search-forward", jedoch können Sie hierbei die Zeichenfolge durch anderen Text ersetzen. Geben Sie nach der Eingabeaufforderung die Zeichenfolge ein, durch die die gefundene Zeichenfolge ersetzt werden soll.
Query-s-r	Esc,Q	Entspricht in der Funktion dem Befehl "Search-replace". Allerdings müssen Sie hier jeweils bestätigen, ob eine Zeichenfolge ersetzt werden soll. Sie haben folgende Optionen zur Auswahl: Y (ja), N (nein), C (alle Zeichenfolgen ersetzen) und Ctrl-G (abbrechen).
Fence-match	Esc, Ctrl-F	Der Text wird nach dem nächsten Zeichen durchsucht, das dem an der aktuellen Cursorposition entspricht. Dies gilt für alle Klammernzeichen (runde, eckige, geschweifte und spitze Klammern). Bei Klammern wird nach derjenigen schließenden Klammer gesucht, die der öffnenden an der aktuellen Cursorposition entspricht.

4.2.3.8 Menü "Extras"

Dieses Menü umfaßt MEmacs-Arbeitsbefehle und Makrobefehle. Bei vielen dieser Befehle muß ein numerisches Argument angegeben werden, bevor Sie den Befehl selbst auswählen; ein * bedeutet, daß ein Argument erforderlich ist. Makrobefehle werden über den Menüpunkt "Execute-macro" ausgeführt.

Set-arg	Ctrl-U	Sie können für einen folgenden Befehl ein numerisches Argument angeben.
Set	Esc,S	Mit diesem Befehl können Sie folgende MEmacs-Parameter einstellen:

- **Screen** – Die MEmacs-Anzeige erfolgt in einem Workbench-Fenster oder auf einem eigenen Bildschirm.
- **Interlace** – Der Zeilensprungmodus wird ein- bzw. ausgeschaltet.
- **Mode** – Sie erhalten erneut die Aufforderung "Mode:". Geben Sie "Cmode" (zum Edieren von C-Programmen) oder "Wrap" (für automatischen Zeilenumbruch) ein. Bei "Cmode" kann der Befehl "Fence-match" automatisch durchgeführt werden. Wenn Sie einen Modus löschen oder hinzufügen wollen, verwenden Sie "-Modus" bzw. "+Modus".
- **Left** – Festlegung des linken Rands. Fordert zur Angabe eines numerischen Arguments auf, wenn es nicht zusammen mit dem Eintrag angegeben wurde.
- **Right** – Festlegung des rechten Rands. Fordert zur Angabe eines numerischen Arguments auf, wenn es nicht zusammen mit dem Eintrag angegeben wurde.

Befehl	Taste	Beschreibung
		Tab — Festlegung des Tabulatorabstands. Fordert zur Angabe eines numerischen Arguments auf, wenn es nicht zusammen mit dem Eintrag angegeben wurde.
		Indent — Festlegung der Einrückung der verschiedenen Verschachtelungsebenen im Modus "Cmode". Fordert zur Angabe eines numerischen Arguments auf, wenn es nicht zusammen mit dem Eintrag angegeben wurde.
		Case — Schaltet die Unterscheidung zwischen Groß- und Kleinschreibung bei Suchbefehlen ein bzw. aus. Die Vorgabe ist "aus".
		Backup — Schaltet die Funktion für Sicherungskopien ein bzw. aus. Sie haben folgende Optionen: ON (die aktuelle Datei wird in <Dateiname>.bak umbenannt und diese Sicherungskopie im Verzeichnis T: gespeichert), SAFE (es wird überprüft, ob die Datei bereits existiert; wenn ja, wird eine Fehlermeldung ausgegeben, und die bestehende Datei wird nicht überschrieben; Ctrl-X löscht den Inhalt der Bildschirmanzeige) und OFF (Vorgabeoption: es wird keine Sicherungskopie erstellt).
Start-macro	Ctrl-X,(	Alle nachfolgenden Eingaben werden in einer Makrodatei gespeichert. Dies ist ein Makrobefehl, der in Verbindung mit den Befehlen "Stop-macro" und "Execute-macro" verwendet wird.
Stop-macro	Ctrl-X,)	MEmacs beendet das Speichern der Tastatureingaben in der Makrodatei.

Execute-macro	Ctrl-X,E	Die zwischen "Start-macro" und "Stop-macro" gemachten Eingaben werden ausgeführt.
Set-key	Ctrl-X, Ctrl-K	Mit diesem Befehl können Sie die Funktionstasten, die umgeschalteten Funktionstasten, die Help-Taste und die Tasten auf dem numerischen Tastenblock als Makrotasten definieren. Das Menükürzel für Ctrl-@ kann nicht verwendet werden, um den Befehl Set-mark in Definitionen von Makrotasten aufzunehmen.
Reset-keys	Esc,K	Die durch den Befehl "Set-key" neu belegten Tasten erhalten wieder ihre vorgegebene Belegung.
Execute-file	Esc,E	Ermöglicht die Ausführung einer Programmdatei innerhalb von MEmacs.
Execute-line	Ctrl-[, Ctrl-[	MEmacs wird in den Befehlsmodus versetzt. Geben Sie hinter der Eingabeaufforderung einen beliebigen Menübefehl in Textform und die zugehörigen Parameter ein. Alternativer Tastaturkurzbefehl: Esc,Esc.

Die folgende Tabelle enthält die vorgegebenen Belegungen für die Funktionstasten (Set-key), wenn sie in Makros vewendet werden.

Taste	Funktion	Tastenfolge
F1	Zeile duplizieren	Ctrl-A,Ctrl-K,Ctrl-Y,Ctrl-M,Ctrl-Y
F2	Zeile löschen	Ctrl-X,Ctrl-D
F3	Tastaturmakro ausführen	Ctrl-X,E
F4	Nächste Seite	Ctrl-V
F5	Vorhergehende Seite	Esc,V
F6	Fenster teilen	Ctrl-X,2
F7	Ein Fenster	Ctrl-X,1
F8	Fenster nach oben rollen	Ctrl-Z

F9	Fenster nach unten rollen	Esc,Z
F10	Datei speichern und verlassen	Ctrl-X,Ctrl-F
Help	Zeile einfügen	Ctrl-J
Enter-Taste (Ziffernblock)	Zeile einfügen	Ctrl-J

4.2.4 Nicht in Menüs enthaltene Befehle

Die folgenden Befehle sind nicht in den Menüs enthalten, Sie können sie also nur über die Tastatur auswählen.

Tasten sind belegt, wenn über sie eine Funktion ausgeführt werden kann. Zum Beispiel ist jede Taste oder Tastenfolge, die als Kurzbefehl für einen Menüpunkt verwendet werden kann, mit der Funktion dieses Menüpunkts belegt.

Taste beschreiben	Esc, Ctrl-D	Über diesen Befehl wird die Belegung einer Taste oder einer Tastenfolge angezeigt. Wenn Sie diesen Befehl auswählen, werden Sie aufgefordert, die entsprechende Tastenfolge einzugeben.
Taste belegen	Esc, Ctrl-B	Mit diesem Befehl können Sie eine Taste mit einer Funktion belegen. Wenn die Eingabeaufforderung erscheint, geben Sie die Funktion und dann die Taste bzw. Tastenfolge ein.
Tastenbelegung aufheben	Esc, Ctrl-U	Mit diesem Befehl können Sie die Belegung einer Taste löschen. Wenn die Eingabeaufforderung erscheint, geben Sie die Taste bzw. Tastenfolge ein, deren Belegung aufgehoben werden soll. Die standardmäßige Belegung der Tasten, die als Befehle verwendet werden, kann nicht gelöscht werden.
Echo	Esc, Ctrl-E	Die Eingabe in die Befehlszeile wird angezeigt.
Zum Fensterrand gehen	Shift+ Cursor	Der Cursor wird an den oberen, unteren, rechten oder linken Bildschirmrand gestellt.

Nächstes Zeichen löschen	Ctrl-D	Das Zeichen an der aktuellen Cursorposition wird gelöscht. Dieser Befehl hat dieselbe Funktion wie die Del-Taste.
Vorheriges Zeichen löschen	Ctrl-H	Das Zeichen links vom Cursor wird gelöscht. Dieser Befehl hat dieselbe Funktion wie die Rücktaste.
Nächste Zeile	Ctrl-M	Hinter der aktuellen Cursorposition wird ein Zeilenvorschubzeichen eingefügt, und der Cursor wird an den Beginn der neuen Zeile gesetzt. (Wie Eingabetaste)
x-Anzahl Zeichen weitergehen	Ctrl-F, Ctrl-B	Mit diesem Befehl kann der Cursor um die angegebene Anzahl von Stellen vorwärts bzw. rückwärts gesetzt werden. Der Vorgabewert ist ein Zeichen.

4.2.5 Individuelle Anpassung von MEmacs

Wenn MEmacs aufgerufen wird, sucht das Programm nach der Datei Emacs_pro, um festzustellen, ob irgendwelche automatisch auszuführende Befehle oder lokale Dateien vorhanden sind. Sie können die Datei Emacs_pro individuell anpassen, indem Sie häufig benötigte Befehle, Befehlsfolgen oder Textzeichenfolgen hinzufügen. Wenn noch keine Datei Emacs_pro vorhanden ist, können Sie selbst eine erstellen.

Wenn Sie eine globale Befehlsdatei erstellen wollen, speichern Sie die Datei "Emacs_pro" im Verzeichnis S:. Lokale Dateien können Sie in ein beliebiges Verzeichnis aufnehmen. Es wird jeweils die lokale Datei in dem Verzeichnis ausgeführt, von dem aus MEmacs aufgerufen wird.

Wenn sowohl eine lokale als auch eine globale Datei Emacs_pro vorhanden ist, hat die lokale Datei Vorrang vor der globalen Datei.

Beispiel:

```
Set Case On
Set-Key F11 "Sehr geehrte Damen und Herren,"
Set-Key F12 "^S Workbench"
Set-Key F13 "^X^B"
```

Dadurch werden folgende Tastenbelegungen definiert:

Shift+F1	Die Zeichenkette "Sehr geehrte Damen und Herren," wird eingegeben.
Shift+F2	Das nächste Vorkommen des Worts "Workbench" wird gesucht (durch den Befehl "Set Case On" wird dabei nach Groß- und Kleinschreibung unterschieden).
Shift+F3	Die Liste der Pufferspeicher wird angezeigt.

Beachten Sie, daß Sie bei der Eingabe von Ctrl-Tastenfolgen Ctrl-Q verwenden müssen. Wenn Sie z. B. "^S" aus diesem Beispiel eingeben wollen, dann tippen Sie "Ctrl-Q Ctrl-S" ein.

4.2.6 Verlassen von MEmacs

Sie können MEmacs durch Auswählen des Menüpunkts "Quit" aus dem Menü "Project" oder durch Eingabe von Ctrl-C verlassen. MEmacs läßt Ihnen die Wahl, entweder vorher modifizierte Puffer zu speichern oder das Programm zu verlassen, ohne zu speichern.

4.3 EDIT

EDIT ist ein Zeileneditor zum automatischen Edieren von Dateien, besonders Binärdateien oder solchen Dateien, die größer sind als der verfügbare Arbeitsspeicher. Mit EDIT können Sie keine neuen Dateien erstellen.

EDIT verarbeitet Dateien zeilenweise. Die einzelnen Zeilen der Eingabe- oder Quelldatei werden nach Änderungen in eine sequentielle Ausgabe- oder Zieldatei übertragen.

Normalerweise verarbeitet EDIT die Quelldatei sequentiell vorwärts. Es kann jedoch auch eine begrenzte Anzahl von Zeilen zurückgegangen werden. Dies ist möglich, da EDIT die Zeilen nicht unmittelbar in die Zieldatei überträgt, sondern sie in einer Ausgabewarteschlange ablegt. Die Größe dieser Warteschlange hängt vom verfügbaren Speicherplatz ab. Wenn Sie die Warteschlange vergrößern wollen, können Sie dies über die Optionen OPT P und OPT W tun.

Der Befehl EDIT hat folgendes Format:

> EDIT [FROM] <dateiname> [[TO] <dateiname>] [*WITH* <dateiname>] [*VER* <dateiname>] [*OPT* P <zeilen> | W <zeichen> | P<zeilen>W<zeichen>] [WIDTH <zeichen>] [PREVIOUS <zeilen>]

Im Argument FROM geben Sie die Quelldatei an, die Sie edieren wollen. Eine Quelldatei muß angegeben werden, obwohl das Schlüsselwort FROM selbst wahlfrei ist.

Mit dem Argument TO geben Sie die Zieldatei an, in die EDIT die geänderten Zeilen übertragen soll. Wenn Sie dieses Argument nicht angeben, wird eine temporäre Datei verwendet. Diese wird nach der als FROM angegebenen Datei benannt und überschreibt sie nach dem Edieren.

Mit dem Schlüsselwort WITH geben Sie eine Datei mit Edierbefehlen an.

Mit dem Schlüsselwort VER (Verifikation) geben Sie die Datei an, in die EDIT Fehlermeldungen und Zeilenüberprüfungen überträgt. Wenn dieses Argument nicht angegeben wird, wird zur Ausgabe der Bildschirm verwendet.

Mit OPT P <n> und OPT W <n> können in anderer Schreibweise die Optionen PREVIOUS und WIDTH (s. u.) angegeben werden. Verwenden Sie aber nicht das Schlüsselwort OPT zusammen mit PREVIOUS und WIDTH.

Mit den Argumenten PREVIOUS und WIDTH können Sie den verfügbaren Speicherplatz vergrößern bzw. verkleinern. Über PREVIOUS geben Sie mit <n> (ganze Zahl) die Anzahl der zwischengespeicherten vorhergehenden Zeilen an. Über WIDTH geben Sie mit <n> die maximale Zeilenlänge in Zeichen an. EDIT multipliziert die Anzahl der vorhergehenden Zeilen mit der Höchstzahl von Zeichen pro Zeile (PREVIOUS * WIDTH), um so den verfügbaren Speicherplatz zu errechnen. Die Vorgaben sind 40 für PREVIOUS und 120 für WIDTH.

4.3.1 Starten von EDIT

Starten Sie EDIT von einer Shell aus mit folgendem Befehl:

```
1.> EDIT <dateiname>
```

Dabei ist <dateiname> der Namen einer existierenden, zu bearbeitenden Datei.

4.3.2 EDIT-Befehle

Die folgende Liste liefert Hintergrundinformationen über die EDIT-Befehle:

Aktuelle Zeile — Die aktuelle Zeile ist diejenige Zeile, die gerade bearbeitet wird. Jeder eingegebene Befehl bezieht sich auf die aktuelle Zeile, Textänderungen erfolgen in der aktuellen Zeile, und neue Zeilen werden vor der aktuellen eingefügt.

Originalzeilen — Die Zeilen der Quelldatei. Zeilen behalten ihre ursprüngliche Numerierung, bis sie mit einem der Befehle REWIND oder = neu numeriert werden.

Nicht-Originalzeilen — Alle Zeilen, die in die Quell- oder Originaldateien eingefügt oder aufgeteilt werden. Diesen werden keine Zeilennummern zugewiesen.

Verifizierung von Zeilen — Wenn Sie Befehle verwenden, die die Daten in einer Zeile verändern, zeigt EDIT bei Verifizierung die geänderte Zeile nach Ausführung des Befehls an.

Argumente — Zeichenfolgen, qualifizierte Zeichenfolgen, Zahlen und Schalterwerte, die zusammen mit EDIT-Befehlen verwendet werden.

Geben Sie Befehle auf eine der drei folgenden Arten ein:

- Befehl eingeben und die Eingabetaste drücken
- Letztes Befehlsargument eingeben und die Eingabetaste drücken
- Ein Semikolon oder eine schließende Klammer eingeben

In den Befehlsbeschreibungen gelten die folgenden Textkonventionen:

- Befehlsnamen stehen in Großbuchstaben, obgleich EDIT nicht zwischen Groß- und Kleinschreibung unterscheidet.

- Spitze Klammern bedeuten, daß Angaben gemacht werden müssen. Beispiel: <Zfolge> bedeutet, daß der Befehl ein Zeichenfolgenargument benötigt.
- Ein <n> steht für ein numerisches Argument.
- Eckige Klammern bedeuten, daß das Argument wahlfrei ist. Beispiel: [<n>] heißt, daß zu dem Befehl ein numerisches Argument angegeben werden kann, aber nicht muß.
- Schrägstriche dienen als Begrenzungszeichen für Zeichenfolgen; zwischen zwei Zeichenfolgen nur einen Schrägstrich eingeben.
- Punkte dienen als Begrenzungszeichen zwischen Dateinamen (hierfür können keine Schrägstriche verwendet werden, da diese als Trennzeichen für Zeichenfolgen verwendet werden).
- Die Namen der Befehle ergeben sich meist als Abkürzung ihres englischen Namens und lassen sich so leichter merken.

4.3.2.1 Auswählen der aktuellen Zeile

Mit den folgenden Befehlen können Sie sich in der Datei bewegen und die aktuelle Zeile auswählen.

Bestimmte Zeilennummer (Move)	M <n>	Gibt durch Eingabe einer Zeilennummer, eines Punktes (erste Zeile) oder eines Sterns (letzte Zeile) als Argument für M eine neue aktuelle Zeile an. Über die Zeilennummer können Sie nur auf Originalzeilen zugreifen.
Nächste Zeile in Quelldatei (Next)	N	Geht eine Zeile weiter. Steht vor N eine Zahl, wird um die entsprechende Anzahl Zeilen weitergesprungen. Wenn Sie den Befehl in der letzten Zeile der Quelldatei eingeben, wird eine zusätzliche Zeile am Ende der Datei erstellt. Wenn Sie dagegen diesen Befehl ausführen wollen, wenn Sie sich bereits in dieser zusätzlichen Zeile befinden, wird eine Fehlermeldung angezeigt.

Vorherige Zeile in Quelldatei (Previous)	P	Geht eine Zeile zurück. Die wiederholte Eingabe von P geht entsprechend mehr Zeilen zurück: Steht vor P eine Zahl, wird um die entsprechende Anzahl Zeilen zurückgegangen. Sie können nur auf eine vorhergehende Zeile zugreifen, die noch nicht in die Zieldatei übertragen wurde. Standardwert sind 40 Zeilen; dieser kann aber über die Option PREVIOUS verändert werden.
Suchen (Find)	F<Zfolge>	Mit dem Befehl können Sie eine aktuelle Zeile auswählen, indem Sie einen Teil des Zeileninhalts angeben.
Rückwärts suchen (Back Find)	B,F<Zfolge>	Mit diesem Befehl können Sie die Quelldatei rückwärts nach einer Zeile durchsuchen, die die angegebene Zeichenfolge enthält.

4.3.2.2 Edieren der aktuellen Zeile

Mit den folgenden Befehlen werden in der aktuellen Zeile neue Daten hinzugefügt oder vorhandene Daten ersetzt.

Einfügen <Zfolge2> nach <Zfolge1>	A <Zfolge1> <Zfolge2>	Fügt <Zfolge2> nach (After) dem ersten Vorkommen von <Zfolge1> ein.
Einfügen <Zfolge2> vor <Zfolge1>	B <Zfolge2> <Zfolge1>	Fügt <Zfolge2> vor (Before) dem ersten Vorkommen von <Zfolge1> ein.
Ersetzen <Zfolge2> für <Zfolge1>	E <Zfolge2> <Zfolge1>	Ersetzt (Exchange) das erste Vorkommen von <Zfolge1> durch <Zfolge2>.

4.3.2.3 Einfügen und Löschen von Zeilen

Mit diesen Befehlen können Sie Zeilen, die nicht zur Originaldatei gehörten, in die Quelldatei einfügen und aus dieser löschen. Es können auch vollständige Dateien in die Quelldatei eingefügt werden.

Eine oder mehrere Zeilen einfügen (Insert)	I [<n>]	Wenn I alleine oder mit einer Zeilenzahl angegeben wird, wird der Text vor der aktuellen Zeile eingefügt. Wenn I mit einem Stern angegeben wird, wird der Text am Dateiende eingefügt. Um das Ende des einzufügenden Texts zu kennzeichnen, drücken Sie die Eingabetaste, dann Z und wieder die Eingabetaste.
Eine oder mehrere Zeilen löschen (Delete)	D [<n>]	Löscht die aktuelle Zeile, wenn keine Argumente angegeben werden. Bei Angabe einer Zeilennummer wird die betreffende Zeile gelöscht. Bei Angabe eines Zeilennummernbereichs werden die betreffenden Zeilen gelöscht; zwischen den Zahlen keine Interpunktionszeichen angeben. Bei Angabe eines Punktes und eines Sterns als Argumente wird alles von der aktuellen Zeile bis zum Ende der Quelldatei gelöscht.
Alle Zeilen bis zur angegebenen Zeichenfolge löschen (Delete Find)	D,F<Zfolge>	Wenn kein Argument eingegeben wird, wird nach der Zeile mit der zuletzt eingegebenen Zeichenfolge gesucht. Bis zu dieser Zeile werden alle Zeilen gelöscht.
Zeilen löschen und ersetzen (Replace)	R [<n>]	Mit diesem Befehl können Sie Zeilen löschen und dann neue einfügen. Die Angabe einer Zeilennummer hinter R bezeichnet eine bestimmte zu ersetzende Zeile.

Abschlußzeichen ändern	Z <Zfolge>	Teilt EDIT mit, daß das Ende von neuem, einzufügenden Text erreicht wurde. Die Eingabe einer Zeichenfolge hinter Z ändert den Standardwert des Abschlußzeichens.
Aktuelle Informationen über EDIT anzeigen	S,H,D	Zeigt gespeicherte Informationen an, z. B. letzte gesuchte Zeichenkfolge, letzter Befehl oder das Abschlußzeichen.
Leerzeichen am Zeilenende ein-/ausschalten	T,R, +l-	Unterdrückt oder aktiviert eventuell vorhandene Leerzeichen am Zeilenende (TRailing space).

4.3.2.4 Edieren von Zeilenfenstern

Normalerweise führt EDIT Funktionen in bezug auf die ganze aktuelle Zeile aus. Sie können die Zeile jedoch in Abschnitte unterteilen, in denen EDIT dann alle nachfolgenden Befehle ausführt. Diese Zeilensegmente werden als Zeilenfenster bezeichnet. Bei der Verwendung von Kennzeichnern ist der Beginn einer Zeile gleichzusetzen mit dem Anfang eines Zeilenfensters.

Wenn EDIT eine aktuelle Zeile verifiziert, wird die Position des Zeilenfensters mit dem Zeichen > unter der Zeile angezeigt. Der Pfeil wird nicht angezeigt, wenn das Zeilenfenster mit dem Anfang der Zeile beginnt.

Mit den folgenden Befehlen können Sie die Position des Pfeils steuern:

>	Eine Stelle nach rechts.
<	Eine Stelle nach links.
PR	Der Pfeil wird auf den Zeilenanfang zurückgesetzt.
PA <Zfolge>	Der Pfeil wird auf das erste Zeichen hinter der angegebenen Zeichenfolge gesetzt.
<Zfolge>	Der Pfeil wird auf das erste Zeichen vor der angegebenen Zeichenfolge gesetzt.

Mit den folgenden Befehlen wird das Zeichen an der aktuellen Pfeilpositon geändert, dann der Pfeil weitergesetzt:

$	Das Zeichen an der aktuellen Pfeilposition wird klein geschrieben, und der Pfeil wird eine Stelle nach rechts gesetzt.
%	Das Zeichen an der aktuellen Pfeilposition wird groß geschrieben, und der Pfeil wird eine Stelle nach rechts gesetzt.
_	Der Befehl _ (Unterstreichung) löscht das Zeichen an der aktuellen Pfeilpostion, wandelt es in ein Leerzeichen um, und der Pfeil wird eine Stelle nach rechts gesetzt.
#	Das Zeichen an der aktuellen Pfeilposition wird gelöscht, und der Rest der Zeile wird um eine Stelle nach links gerückt. Wenn Sie mehrere Zeichen löschen wollen, geben Sie vor # eine Zahl an. Beispiel: **5#** löscht die nächsten fünf Zeichen im Fenster.

Sie können diese Befehle zum zeichenweisen Edieren der Zeile auch kombinieren.

Mit einigen anderen Befehlen können Sie Text in die aktuelle Zeile einfügen und ersetzen, ähnlich den Befehlen A, B und E (siehe oben). Wenn der Vorgang beendet ist, wird der Pfeil weitergesetzt.

Einfügen <Zfolge2> nach <Zfolge1>	A,P <Zfolge1> <Zfolge2>	Fügt <Zfolge2> nach dem ersten Vorkommen von <Zfolge1> ein. Der Pfeil wird dann hinter <Zfolge2> gesetzt.
Einfügen <Zfolge2> vor <Zfolge1>	B,P <Zfolge1> <Zfolge2>	Fügt <Zfolge2> vor dem ersten Vorkommen von <Zfolge1> ein. Der Pfeil wird dann hinter <Zfolge2> gesetzt.
Ersetzen <Zfolge2> für <Zfolge1>	E,P, <Zfolge1> <Zfolge2>	Ersetzt das erste Vorkommen von <Zfolge1> durch <Zfolge2>. Der Pfeil wird dann hinter <Zfolge2> gesetzt.
Löschen bis Ende (Delete Till After)	D,T,A <Zfolge>	Der Text wird ab Zeilenanfang bzw. ab dem Pfeil bis zum Ende der angegebenen Zeichenfolge gelöscht.
Löschen bis Anfang (Delete Till Before)	D,T,B <Zfolge>	Der Text wird ab Zeilenanfang bzw. ab dem Pfeil bis zum Beginn der angegebenen Zeichenfolge gelöscht.
Löschen ab Ende (Delete From After)	D,F,A <Zfolge>	Der Text wird ab dem Ende der angegebenen Zeichenfolge bis zum Zeilenende gelöscht.

Löschen ab Anfang (Delete From Before)	D,F,B <Zfolge>	Der Text wird ab der angegebenen Zeichenfolge bis zum Zeilenende gelöscht.

4.3.2.5 Trennen und Verbinden von Zeilen

Mit den in diesem Abschnitt erläuterten Befehlen können Sie eine Zeile trennen oder zwei oder mehr Zeilen verbinden.

Zeile vor <Zfolge> trennen (Split Before)	S,B <Zfolge>	Die aktuelle Zeile wird vor der angegebenen Zeichenfolge getrennt. Der erste Teil der Zeile wird an die Ausgabewarteschlange gesendet, der Rest der Zeile wird zur aktuellen Zeile (keine Originalzeile). Mit diesem Befehl können auch Kennzeichner verwendet werden, um die Umgebung der Zeichenfolge einzuschränken.
Zeile nach <Zfolge> trennen (Split After)	S,A <Zfolge>	Die aktuelle Zeile wird nach der angegebenen Zeichenfolge getrennt. Der erste Teil der Zeile wird an die Ausgabewarteschlange gesendet, der Rest der Zeile wird zur aktuellen Zeile. Mit diesem Befehl können auch Kennzeichner verwendet werden, um die Umgebung der Zeichenfolge einzuschränken.
Zwei Zeilen verbinden (Concatenate Lines)	C,L [<Zfolge>]	Die aktuellen Zeile wird mit der nächsten verbunden. Das Argument <Zeichenfolge> muß nicht angegeben werden. Bei Angabe einer Zeichenfolge wird sie an die aktuelle Zeile angefügt, die dann mit der nächsten Zeile der Quelldatei verbunden wird.

4.3.2.6 Neunumerierung von Zeilen

Mit diesen Befehlen können Sie die Zeilen in der Quelldatei neu numerieren, damit Sie neue Zeilen aufnehmen und Dateien aktualisieren können, die ediert wurden.

Quellzeilen neu numerieren	= <n>	Die Nummer der aktuellen Zeile wird auf <n> gesetzt. Wenn Sie dann auf Zeilen unter <n> springen, werden alle folgenden Original- und geänderten Zeilen neu numeriert.
Zurück an den Anfang der Quelldatei	REWIND	Zeile 1 wird zur aktuellen Zeile. EDIT liest dann den Rest der Quelldatei und überträgt die Zeilen in die Zieldatei. Danach wird die Zieldatei geschlossen und als neue Quelldatei wieder geöffnet. Jede geänderte Zeile wird dann wie eine Originalzeile verarbeitet. Kann als REWI eingegeben werden.

4.3.2.7 Verifizieren von Zeilen

Diese Befehle bieten verschiedene Mögkichkeiten, Zeilen zu verifizieren.

Verifizierung ein-/ausschalten	V + I -	Mit diesem Befehl können Sie die Zeilenverifikation ein- und ausschalten. Ausgeschaltet werden die Zeilen nicht angezeigt. Zum Ausschalten V -, zum Einschalten V + eingeben.
Verifizieren der aktuellen Zeile	?	Mit diesem Befehl können Sie die aktuelle Zeile verifizieren, wobei Nummer und Inhalt der Zeile angezeigt werden.

Verifizieren der aktuellen Zeile mit Textanzeigern	!	Zwei Verifizierungszeilen werden angezeigt. In der ersten werden alle nichtgrafischen Zeichen durch das erste Zeichen des zugehörigen Hexadezimalwerts ersetzt. In der zweiten werden unter allen Stellen, die Großbuchstaben entsprechen, ein Minuszeichen und an den Stellen, die nichtgrafischen Zeichen entsprechen, jeweils die zweite Hexadezimalziffer angezeigt. Die anderen Stellen sind mit Leerzeichen besetzt. In binären Dateien werden nichtgrafische Zeichen mit zwei Fragezeichen (??) dargestellt.

4.3.2.8 Überprüfung der Quelldatei

Mit den in diesem Abschnitt erläuterten Befehlen bewegen Sie sich durch die Quelldatei. Dabei werden die übersprungenen Zeilen an die Verifizierungsdatei und an die normale Ausgabe gesendet. Diese Befehle werden als Anzeigebefehle bezeichnet, da Sie sich mit ihnen Zeilen anzeigen lassen können.

<n> Zeilen anzeigen (Type)	T<n>	Die angegeben Anzahl Zeilen wird angezeigt. Die erste Zeile ist die aktuelle. Wenn Sie <n> nicht angeben, wird die Quelldatei bis zum Ende angezeigt. Sie können diesen Befehl mit Ctrl-C abbrechen.
Zeilen in Ausgabewarteschlange anzeigen	T,P	Die derzeit in der Ausgabewarteschlange stehenden Zeilen werden angezeigt.
Zeilen anzeigen, bis alle Zeilen in der Ausgabewarteschlange ersetzt sind	T,N	Ab der aktuellen Zeile werden so lange Zeilen angezeigt, bis alle Zeilen in der Ausgabewarteschlange ersetzt sind. Der bisherige Inhalt wird in die Zieldatei übertragen.

Zeile mit Zeilennummer anzeigen	T,L <n>	Dieser Befehl ist dem Befehl T ähnlich. Er zeigt die angegebene Anzahl Zeilen einschließlich Zeilennummer an. Da eingefügte und geteilte Zeilen keine Nummern haben, zeigt EDIT statt dessen ++++ an.

4.3.2.9 Globale Änderungen

Mit den in diesem Abschnitt erläuterten Befehlen können globale Änderungen gestartet und abgebrochen werden. Beim Durchsuchen der Quelldatei in Vorwärtsrichtung werden die globalen Änderungen automatisch vorgenommen. In den folgenden Befehlen werden automatisch die Befehle A, B oder E entsprechend dem Verwendungszweck auf alle Vorkommen der <Zeichenfolge1> in der jeweiligen aktuellen Zeile angewendet. Ebenso werden sie für die Zeile durchgeführt, die zum Zeitpunkt der Befehlseingabe aktuell war.

```
GA [Kennzeichner] <Zeichenfolge1> <Zeichenfolge2>
GB [Kennzeichner] <Zeichenfolge1> <Zeichenfolge2>
GE [Kennzeichner] <Zeichenfolge1> <Zeichenfolge2>
```

Wenn Sie beispielsweise in der gesamten Datei "DF0:" in "DF2:" ändern wollen, geben Sie folgenden Befehl ein:

```
GE /DF0:/DF2:/
```

Globalen Befehl abbrechen	CG [<ID-Nr.>]	Ein globaler Befehl wird abgebrochen. Die mit einem GA-, GB- oder GE-Befehl angegebene Identifikationsnummer wird an die Verifizierungsdatei gesendet (bzw. an den Bildschirm, wenn EDIT interaktiv ist). Wenn kein Argument eingegeben wird, werden alle globalen Operationen abgebrochen. Wenn Sie eine bestimmte Operation abbrechen wollen, geben Sie hinter dem Befehl CG die Identifikationsnummer an.

Globalen Befehl unterbrechen	SG [<ID-Nr.>]	Unterbricht einen globalen Befehl. Wenn kein Argument eingegeben wird, werden alle globalen Operationen unterbrochen. Wenn Sie eine bestimmte Operation unterbrechen wollen, geben Sie die Identifikationsnummer an.
Globalen Befehl fortsetzen	EG [<ID-Nr.>]	Setzt einen globalen Befehl fort, der mit dem Befehl SG unterbrochen wurde. Wenn kein Argument eingegeben wird, werden alle globalen Operationen fortgesetzt. Ohne Angabe einer spezifischen Identifikationsnummer werden alle globalen Änderungen wieder aufgenommen.
Globale Befehle anzeigen	SHG	Die aktuellen globalen Befehle und die zugehörigen Identifikationsnummern werden angezeigt. Es wird auch angegeben, wie viele Entsprechungen bei der Suche nach einer Zeichenfolge gefunden wurden.

4.3.2.10 Wechseln von Befehls-, Eingabe- und Ausgabedateien

Mit den in diesem Abschnitt erläuterten Befehlen können Sie die Dateien wechseln, die Sie beim Starten von EDIT über die Shell angegeben haben. Hierbei handelt es sich um folgende Dateien:

- Befehlsdatei — sie wird mit der Option WITH gestartet
- Eingabedatei — die als FROM angegebene Quelldatei
- Ausgabedatei — die als TO angegebene Zieldatei.

Ändern der Befehlsdatei	C <dateiname>	Liest EDIT-Befehle aus der angegebenen Datei. Da in AmigaDOS Schrägstriche (/) verwendet werden, um Verzeichnis- und Dateinamen voneinander zu trennen, verwenden Sie dazu ein anderes Zeichen. Wenn EDIT alle Befehle ausgeführt hat, wird die Datei geschlossen. Sie können dann Befehle über die Tastatur eingeben .
Ändern der Eingabedatei	FROM <dateiname>	Liest Zeilen aus einer anderen Quelldatei ein. EDIT schließt die ursprüngliche Quelldatei nicht. Sie können die Quelldatei wieder mit dem Befehl FROM auswählen, ohne ein Argument anzugeben. Siehe Seite 4-63.
Schließen einer Datei	CF <dateiname>	Schließt die Zieldatei, die mit dem Befehl TO angegeben wurde. Sie können dann diese Datei als Eingabedatei öffnen. Mit CF können Sie auch eine neue geöffnete Eingabedatei wieder schließen. Beispiele zum Befehl CF sind auf Seite 4-63 erläutert.
Ändern der Ausgabedatei	TO <dateiname>	Gibt eine andere Zieldatei an. Mit dem Befehl TO wird der Inhalt der Ausgabewarteschlange in der nach TO angegebenen Datei abgelegt. Die nach TO angegebene Datei wird so lange verwendet, bis eine neue angegeben wird. Wenn Sie die ursprüngliche Zieldatei wieder verwenden wollen, geben Sie den Befehl TO ohne Argumente ein. Die andere Ausgabedatei bleibt dann geöffnet, obwohl sie nicht verwendet wird. Beispiele zum Befehl TO sind auf Seite 4-63 erläutert.

Abbrechen der Befehlsdatei	Q	Die Ausführung der Datei, die mit dem Schlüsselwort WITH oder dem Befehl C angegeben wurde, wird abgebrochen. EDIT kehrt zu einer vorhergehenden Befehlsdatei zurück. Die Verwendung des Befehls Q auf der äußersten Ebene entspricht dem Befehl W.

Beispiele für die Verwendung der Befehle FROM, CF und TO:

Befehl	Funktion
M10	Die Zeilen 1-9 der Quelldatei werden in die Ausgabewarteschlange übertragen.
FROM .XYZ.	Die Datei XYZ wird als neue Eingabedatei ausgewählt. Zeile 10 der Quelldatei bleibt die aktuelle Zeile.
M6	Zeile 10 der Quelldatei und dann die Zeilen 1-5 aus der Datei XYZ werden in die Ausgabewarteschlange übertragen. Zeile 6 der Datei XYZ ist die neue aktuelle Zeile.
From	Die Quelldatei wird erneut ausgewählt.
M14	Zeile 6 der Datei XYZ und dann die Zeilen 11-13 der Quelldatei werden in die Ausgabewarteschlange übertragen. Zeile 14 der Quelldatei ist die neue aktuelle Zeile.
FROM .XYZ.	Die Datei XYZ wird erneut ausgewählt. Zeile 14 der Quelldatei ist weiterhin die aktuelle Zeile.
M*	Zeile 14 der Quelldatei und alle verbleibenden Zeilen der Datei XYZ werden in die Ausgabewarteschlange übertragen. Der Datei XYZ wird eine zusätzliche Zeile angefügt; diese ist die neue aktuelle Zeile
FROM	Die Quelldatei wird erneut ausgewählt. Die der Datei XYZ hinzugefügte Zeile bleibt die aktuelle Zeile.
CF .XYZ.	Die Datei XYZ wird geschlossen.
M*	Die verbleibenden Zeilen der Quelldatei (Zeile 15 bis zum Dateiende) werden in die Ausgabewarteschlange übertragen.

Befehl (Forts.)	Funktion (Forts.)
M11	Die Zeilen 1-10 werden von der Quelldatei in die ursprüngliche Zieldatei übertragen.
TO .XYZ.	Die Datei XYZ wird zur neuen Ausgabedatei.
M21	Die Zeilen 11-20 werden in die Datei XYZ übertragen.
TO M31	Die ursprüngliche Zieldatei wird zur aktuellen Datei, und die Zeilen 21-30 werden in sie übertragen.
TO .XYZ.	XYZ wird zur aktuellen Ausgabedatei.
M41	Die Zeilen 31-40 werden in die Datei XYZ übertragen.
TO	Die ursprüngliche Zieldatei wird zur aktuellen Datei.
TO .XYZ.	Die Ausgabewarteschlange wird in die Datei XYZ übertragen.
1000N	Die nächsten 1000 Zeilen in der Quelldatei werden übertragen.
TO	Die ursprüngliche Zieldatei wird ausgewählt.
CF .XYZ.	Die Datei XYZ wird geschlossen.
I2000 .XYZ.	Die 1000 Zeilen aus der Quelldatei, die in die Datei XYZ gesendet wurden, werden wieder in die Quelldatei über Zeile 2000 eingefügt.

4.3.3 Verlassen von EDIT

Mit folgenden Befehlen können Sie EDIT verlassen:

Speichern und Verlassen	W	EDIT wird verlassen, und alle vorgenommenen Änderungen an der durch TO angegebenen Zieldatei werden gespeichert. EDIT wird verlassen, wenn das Ende der Quelldatei erreicht ist, alle Dateien geschlossen sind und der Speicher freigegeben ist. Wenn Sie EDIT gestartet haben, ohne eine Zieldatei anzugeben, erhält die temporäre Zieldatei den gleichen Namen wie die ursprüngliche Quelldatei. Die ursprüngliche Quelldatei erhält den neuen Namen ":T/Edit-backup". Diese Sicherungsdatei ist nur so lange verfügbar, bis Sie das nächste Mal EDIT aufrufen.
Verlassen ohne Speichern	STOP	EDIT wird sofort beendet, ohne Änderungen in der Quelldatei zu speichern. Dadurch wird verhindert, daß die ursprüngliche Quelldatei mit Änderungen überschrieben wird.



Kapitel 5

Befehlsdateien/Skripts

Befehlsdateien oder Skripts sind Textdateien, die Listen von Befehlen zur Ausführung sich wiederholender oder komplizierter Aufgaben oder ein und derselben Operation für mehrere Dateien enthalten. Dieses Kapitel enthält die folgenden Informationen über Skript-Dateien:

- Was sind Skripts?
- Skript-Sonderzeichen
- Skript-Befehle
- Bedingungskennzeichen
- Testhilfe/Fehlersuche in Skripts
- Umgebungsvariablen

5.1 Was sind Skripts?

Ein Skript ist eine Textdatei, die eine Reihe von Befehlen enthält. Die Verwendung von Skripts ist eine Möglichkeit, komplexe oder sich wiederholende Aufgaben zu automatisieren, besonders solche, die Sie in regelmäßigen Abständen ausführen müssen. Ein Skript kann praktisch jede Operation übernehmen, die ansonsten aus mehreren Einzelbefehlen besteht, einschließlich des Arbeitens mit Programmen und Datendateien, des Ausführens von Berechnungen und des Dialogbetriebs, d. h. des Empfangens und Anzeigens von Informationen. Im Grunde ist ein Skript einfach ein kleines, leicht zu edierendes Programm.

Zum Erstellen eines Skripts gehen Sie wie folgt vor:

1. Öffnen Sie einen Texteditor, z. B. ED, der Dateien im ASCII-Format speichert.
2. Geben Sie im Texteditor die Skript-Befehle in der Reihenfolge ein, in der sie ausgeführt werden sollen.
3. Speichern Sie die Datei. Normalerweise wird das Verzeichnis S: für Skript-Dateien verwendet, Sie können aber auch an einer beliebigen anderen Stelle abspeichern.

Danach können Sie das Skript in der Shell aufrufen, indem Sie den Befehl EXECUTE gefolgt vom vollständigen Pfadnamen der Skript-Datei eingeben.

Hinweis Die Angabe des Befehls EXECUTE und des vollständigen Pfades können Sie sich sparen, indem Sie das s-Schutzbit des Skripts setzen. Geben Sie den Befehl **PROTECT <script> +s** ein, unter Angabe des Pfadnamens der Skript-Datei. Wenn das s-Bit gesetzt ist, müssen Sie zum Aufrufen des Skripts nur seinen Namen eingeben.

Es ist möglich, ein Skript per Maus-Doppelklick aufzurufen, das ein Workbench-Projektpiktogramm besitzt, in dem Sie den Befehl ICONX als Standardprogramm eintragen. Näheres zum Befehl ICONX siehe in Kapitel 6.

5.1.1 Verschiedene Arten von Skripts

Auf dem Amiga können verschiedene Arten von Skripts verwendet werden, und zwar solche mit AmigaDOS-Befehlen, Skripts mit ARexx-Befehlen und Skripts mit Editor-Befehlen (ED, MEmacs und EDIT). Näheres zum Schreiben von ARexx-Skripts finden Sie im *ARexx Benutzerhandbuch*. Informationen über Editor-Befehlsskripts finden Sie in Kapitel 4.

5.1.1.1 Wann wird ARexx verwendet?

Sie können auf dem Amiga sowohl AmigaDOS- als auch ARexx-Skripts erstellen. Diese Skripts schließen sich nicht gegenseitig aus; unter AmigaDOS Version 2 und höher dürfen ARexx-Befehle in AmigaDOS-Skripts verwendet werden und umgekehrt. Allerdings eignen sich die verschiedenen Skript-Typen jeweils besser für bestimmte, typenspezifische Aufgaben.

Im Gegensatz zu ARexx ist AmigaDOS keine allgemeine Programmiersprache. AmigaDOS ist für grundlegende Aufgaben der Dateiverwaltung und Systemkonfiguration ausgelegt. ARexx dient zum Ausführen komplexerer Programme als einfachen Verzweigungen und bedingter Ausführung von Befehlen.

5.1.1.2 Einfache Skripts

Ein Skript kann einfach aus mehreren Pfadangaben bestehen, in denen der Amiga angewiesen wird, bestimmte Programme auszuführen. Eine solche einfache Skript-Datei ist z. B. die Datei User-startup. In dieser Datei haben Sie die Möglichkeit, verschiedene Konfigurationsbefehle einzufügen wie etwa die Anweisung ADDBUFFERS, ohne daß Gefahr für den Programmfluß besteht oder eine Fehlerprüfung vonnöten wäre. Näheres zur Datei User-startup finden Sie in Anhang D.

5.1.1.3 Automatische Skripts

Mit dem Befehl LIST können Sie Skripts automatisch generieren. LIST besitzt eine Option LFORMAT, über die Sie die Befehlsausgabe so verändern können, daß Sie jeden Text als Ausgabe erhalten können, den Sie zusammen mit der üblichen Ausgabe von LIST haben möchten. Dieser Text kann ein Befehl mit Befehlsschlüsselwörtern sein, wobei der von LIST ausgegeben Dateiname als Argument zum Befehl erscheint. Wenn Sie den Inhalt eines Verzeichnisses auf diese Weise auflisten und dabei die Ausgabe in eine Datei umleiten, haben Sie bereits ein fertiges Skript, das einen Befehl auf den Inhalt dieses Verzeichnisses anwenden läßt.

Wenn ein Befehl für mehrere Dateien zu kompliziert ist, um mit einem einzigen Suchmusterabgleichbefehl ausgeführt zu werden, können Sie ein automatisch generiertes Skript verwenden, um den Befehl für jede einzelne der Dateien einzeln aufzurufen. Beispiel: eine Operation, bei der eine Reihe von Dateien umbenannt wird und bei der die aktuellen Dateinamen und die gleiche Erweiterung angegeben werden, kann nicht von einem einzelnen RENAME-Befehl mit Suchmusterabgleich erledigt werden. Im Abschnitt LIST in Kapitel 6 finden Sie weitere Informationen über die LIST-Option LFORMAT, am Ende von Kapitel 8 sehen Sie ein ausführliches Beispiel dieses Verfahrens.

5.2 Spezielle Skript-Zeichen

Das Semikolon, das umgekehrte Hochkomma, das Dollarzeichen, das doppelte Dollarzeichen und das Fragezeichen sind Sonderzeichen, die hauptsächlich in Skript-Dateien zum Hinzufügen von Kommentaren, zum Aufrufen von Befehlen innerhalb von Zeichenfolgen, zum Einleiten von Umgebungsvariablen, zur Bezugnahme auf die aktuelle Shell-Nummer und zum Akzeptieren der Eingabeumleitung verwendet werden.

Semikolon (;)

Semikolons fügen Kommentare zu Befehlszeilen hinzu. Alle Zeichen rechts von einem Semikolon werden ignoriert, Sie können also beschreibende Kommentare in die gleiche Zeile wie AmigaDOS-Befehle schreiben. Beispiel:

```
ASSIGN T: RAM:t      ;Einrichten von Verzeichnis T:
                     ;für temporäre Dateien
```

Kommentare können auch in Folgezeilen fortgesetzt werden, wenn sie nicht in eine Zeile passen. Neue Zeilen müssen wieder mit einem Semikolon beginnen und sollten aus Gründen der Übersichtlichkeit genausoweit eingerückt sein wie die vorherige Kommentarzeile.

Umgekehrtes Hochkomma (`)

Umgekehrte Hochkommas dienen zum Aufrufen von Befehlen innerhalb einer Zeichenfolge. Wenn eine Zeichenfolge ausgegeben wird, die einen zwischen umgekehrten Hochkommas stehenden Befehl enthält, wird dieser Befehl ausgeführt. Beispiel:

```
1> ECHO "Datum und Uhrzeit sind: `date`"
```

gibt "Datum und Uhrzeit sind: " gefolgt von der Ausgabe des Befehls DATE aus. Wenn ein Befehl wie etwa DIR, der mehrzeilige Ausgaben produziert, in eine ECHO-Anweisung eingefügt ist, wird die Ausgabe nicht korrekt formatiert; sie erscheint dann in einer einzigen Zeile.

Hinweis — Befehle, die sich auf das aktuelle Verzeichnis beziehen, funktionieren nicht einwandfrei, wenn sie zwischen umgekehrten Hochkommas stehen und innerhalb einer Zeichenfolge aufgerufen werden. Das umgekehrte Hochkomma richtet automatisch eine temporäre, nur für diesen Befehl bestimmte Sub-Shell ein. Bezugnahmen auf ein aktuelles Verzeichnis greifen aber auf das Verzeichnis der Sub-Shell zu.

Dollarzeichen ($)

Das Dollarzeichen wird auf zwei Arten verwendet: als Operator, der eine Umgebungsvariable einleitet (dies ist auch außerhalb eines Skripts möglich), und in zwischen spitzen Klammern stehenden Befehlsteilen zur Trennung eines variablen Werts von einem Standardwert.

Beispiel mit einer Umgebungsvariable:

```
1> ECHO "Aktuelle Kickstart-Version: $Kickstart"
Aktuelle Kickstart-Version: 40.68
```

Als Standard-Trennzeichen in einem Skript:

```
COPY foo.library TO <LIBS:$userlibdir>
```

Hierbei sei userlibdir ein in einer .KEY-Zeile (s. u.) definiertes Argument. Wenn es beim Aufruf nicht angegeben wurde, wird an dieser Stelle dann der Standardwert LIBS: verwendet.

Mit Hilfe des Skript-Schlüsselworts .DOLLAR (s. u.) können Sie diese Funktion auch mit einem anderen Zeichen als dem Dollarzeichen belegen.

Doppeltes Dollarzeichen (<$$>)

Zwei in spitzen Klammern stehende Dollarzeichen (<$$>) ersetzen die aktuelle Prozeßnummer. Auf die aktuelle Shell-Nummer kann mit der Zeichenfolge "doppeltes Dollarzeichen" <$$> einschließlich der Klammern Bezug genommen werden, da damit stets die aktuelle Prozeßnummer als Zeichenfolge zurückgemeldet wird. Wenn Sie temporäre Dateien in einer Multitasking-Umgebung erstellen, ist es entscheidend, daß diese Dateien eindeutige Namen haben, damit keine Prozesse miteinander in Konflikt geraten. Das Hinzufügen der Zeichenfolge <$$> zu Dateinamen erstellt eindeutige Dateinamen für temporäre Dateien, logische Zuordnungen und PIPEs. Eine Anweisung .KEY ist in jedem Skript erforderlich, in dem <$$> verwendet wird. .KEY wird auf Seite 5-9 beschrieben. Zur Vermeidung von Konflikten mit eventuellen Umleitungsargumenten können die spitzen Klammern von <$$> mit Hilfe der Befehle .BRA und .KET umdefiniert werden, meist in geschweifte Klammern. .BRA und .KET werden auf Seite 5-10 beschrieben.

Fragezeichen (?)

Das Fragezeichen, wenn es als separates Argument in einem Befehl innerhalb eines Skripts verwendet wird, weist den Befehl an, Eingabeumleitung zu akzeptieren.

5.3 Skript-Befehle

Jeder AmigaDOS-Befehl kann in einem Skript verwendet werden, es gibt aber einige Befehle, die ausschließlich in Skripts verwendet werden. Mit Skript-Befehlen sind folgende Operationen möglich:

- Parametersubstitution
- Eingabe von Argumenten in die Befehlszeile
- E/A-Umleitung
- Vorgabe von Standardzeichenfolgen

- Eingabe von Kommentaren
- Verschachtelung von Befehlen in Skripts
- Erstellen interaktiver Skripts
- Erstellen von Skripts, die Befehle wiederholen

5.3.1 Skriptspezifische Befehle

Die folgenden Befehle werden normalerweise nur in Skripts verwendet:

Skript-Befehl	Bedeutung
ASK	Fordert Benutzereingabe an.
ECHO	Gibt eine Zeichenfolge aus.
ELSE	Ermöglicht eine Alternative in einem bedingten Block.
ENDIF	Beendet einen IF-Block.
ENDSKIP	Beendet einen SKIP-Block.
EXECUTE	Führt ein Skript mit wahlfreier Argumentsubstitution aus.
FAILAT	Setzt die Fehlerabbruchgrenze des Skripts.
IF	Bearbeitet bedingte Operationen.
LAB	Gibt eine Sprungmarke an; wird zusammen mit SKIP verwendet.
QUIT	Gibt einen Rückkehrcode an und verläßt ein Skript.
REQUESTCHOICE	Gibt AmigaDOS- und ARexx-Skripts die Möglichkeit, Systemdialogfenster zu benutzen, um Benutzerantworten anzufordern.
REQUESTFILE	Gibt AmigaDOS- und ARexx-Skripts die Möglichkeit, das Dateiauswahlfenster des Systems zu benutzen.
SKIP	Überspringt die Ausführung des Skript vorwärts oder rückwärts zur angegebenen Sprungmarke.
WAIT	Wartet die angegebene Zeit.

5.3.2 Punkt-Befehle

Punkt-Befehle sind Schlüsselwörter, die mit einem Punkt beginnen; sie werden ausschließlich in Skripts verwendet. Spezielle Punkt-Befehlszeilen in einem Skript können z. B. Stellen für die Parametersubstitution angeben. Geben Sie diese Parameter später als Argumente zum Befehl EXECUTE ein. Die folgende Tabelle zeigt die verschiedenen Punkt-Befehle.

Punkt-Befehle	Bedeutung
.KEY	Argumentschablone zur Angabe des Formats von Argumenten; kann als .K abgekürzt werden. Trennen Sie Argumente zu .KEY mit Kommas voneinander. Verwenden Sie keine Leerzeichen. Weitere Informationen zur Verwendung von .KEY finden Sie in Kap. 5.3.2.1.
.DOT <z>	Ändert das Punkt-Zeichen von . auf <z>, d. h. <z> ist das neue, zu verwendende Zeichen.
.BRA <z>	Ändert das Zeichen für die öffnende spitze Klammer von < auf <z>.
.KET <z>	Ändert das Zeichen für die schließende spitze Klammer von > auf <z>.
.DOLLAR <z>	Ändert das Standardzeichen von $ auf <z>; kann als .DOL abgekürzt werden.
.DEF <Schlüsselwort> <Wert>	Vergibt den Standardwert an den Parameter.
.<Leerzeichen>	Kommentarzeile. Achten Sie darauf, nach dem Punkt ein Leerzeichen einzugeben, um Fehler zu vermeiden. Die empfohlene Methode der Kommentareingabe ist diejenige mit dem Semikolon (;).
.	Leere Kommentarzeile. Achten Sie darauf, daß außer dem Punkt nichts in der Zeile steht, um Fehler zu vermeiden.

Wenn eine Befehlszeile mit EXECUTE aufgerufen wird, liest AmigaDOS die erste Zeile der Skript-Datei. Wenn sie mit einem Punkt-Befehl beginnt, durchsucht AmigaDOS das Skript nach den

oben beschriebenen Parametersubstitutionen und erstellt eine temporäre Datei im Verzeichnis T:. Wenn die Datei nicht mit einem Punkt-Befehl beginnt, geht AmigaDOS davon aus, daß keine Parametersubstitution erforderlich ist, und beginnt unmittelbar mit der Ausführung der Datei, ohne sie nach T: zu kopieren. Da Punkt-Befehle zusätzliche Disk-Zugriffe erforderlich machen und damit die Ausführungszeit erhöhen, setzen Sie diese Befehle nur dann ein, wenn Sie auf die Parametersubstitution nicht verzichten können.

5.3.2.1 Zulassen von Argumenten

Das Schlüsselwort .KEY (oder .K) gibt Schlüsselwortnamen und -positionen in der Befehlszeile an. Es teilt EXECUTE mit, wie viele Parameter vorhanden sind und wie sie zu interpretieren sind. Pro Skript ist nur eine .KEY-Anweisung zulässig; wenn sie verwendet wird, sollte sie in der ersten Zeile der Datei stehen. Jedes Skript, das <$$> enthält, muß auch eine .KEY-Anweisung enthalten.

Die Argumente in der .KEY-Zeile können mit den Zusätzen /A und /K versehen werden, die die gleiche Funktion haben wie in einer AmigaDOS-Schablone. (Schablonen werden in Kapitel 6 beschrieben.) Argumente, auf die ein /A folgt, sind erforderlich, Argumente, auf die ein /K folgt, erfordern den Namen des betreffenden Arguments als Schlüsselwort. Verwenden Sie Kommas (keine Leerzeichen) als Trennzeichen zwischen mehreren Argumenten in der .KEY-Zeile.

Beispiel: Wenn ein Skript mit .KEY Dateiname/A beginnt, bedeutet dies, daß ein Dateiname in der EXECUTE-Befehlszeile nach dem Namen des Skripts angegeben werden muß. Dieser Dateiname wird in den folgenden Zeilen des Skripts substituiert. Beispiel: Wenn Ihr Skript "Neutext" heißt und die erste Zeile

```
.KEY Dateiname/A,TO/K
```

lautet, muß eine Dateinamenvariable angegeben werden. Die Variable TO ist wahlfrei, wenn sie aber angegeben wird, muß das Schlüsselwort TO verwendet werden. Folgendes wäre eine akzeptable Art des Aufrufs von "Neutext":

```
1> EXECUTE Neutext Textdatei TO NeuDatei
```

5.3.2.2 Substitution

Vor der Ausführung durchsucht AmigaDOS das Skript nach Angaben zwischen den Zeichen .BRA und .KET (standardmäßig < und >). Solche Angaben können aus einem Schlüsselwort bzw. einem Schlüsselwort und einem Standardwert bestehen. EXECUTE versucht, einen Parameter zu substituieren, wenn das Programm ein Schlüsselwort in spitzen Klammern findet. Wenn Sie allerdings eine Zeichenfolge in Ihrem Skript verwenden möchten, die spitze Klammern enthält, oder wenn Ihr Skript spitze Klammern zur Umleitung von Ein-/Ausgabe verwendet, müssen Sie Ersatzzeichen für die spitzen Klammern definieren; dazu dienen die Befehle .BRA und .KET. .BRA <z> ändert die öffnende Klammer in das in <z> angegebene Zeichen, .KET <z> nimmt die entsprechende Änderung der schließenden Klammer vor.

Beispiel: Das folgende Skript "Demo" verwendet folgende Zeilen:

```
.KEY Dateiname
ECHO "Dies gibt KEINE <spitzen> Klammern aus."
.BRA {
.KET }
ECHO "Diese Zeile GIBT <spitze> Klammern aus."
ECHO "Der angegebene Dateiname ist {Dateiname}."
```

1> EXECUTE Demo TestDatei

Dies führt zu folgender Ausgabe:

```
Dies gibt KEINE Klammern aus.
Diese Zeile GIBT <spitze> Klammern aus.
Der angegebene Dateiname ist TestDatei.
```

Die erste ECHO-Anweisung läßt AmigaDOS nach einer Variable suchen, die für den Parameter <spitzen> zu substituieren ist. Da kein Argument mit Namen "spitzen" in der EXECUTE-Befehlszeile angegeben wird, wird die Nullzeichenfolge substituiert. Die Befehle .BRA und .KET weisen das Skript an, zum Einschließen von Parametern keine spitzen, sondern geschweifte Klammern zu verwenden. Wenn die zweite ECHO-Anweisung ausgeführt wird, werden die spitzen Klammern - ohne ihre frühere Sonderbedeutung - als normaler Text ausgegeben. Die dritte ECHO-Anweisung zeigt, daß die geschweiften Klammern nun die Funktion der spitzen Klammern übernommen haben.

Zur Vermeidung von Konflikten bei der Umleitung sind die spitzen Klammern in <$$> mit den Befehlen .BRA und .KET neu zu definieren.

5.3.2.3 Standardwerte

Wenn Sie ein Schlüsselwort zwischen spitze Klammern schreiben, können Sie auch eine Standardzeichenfolge angeben, die verwendet werden soll, wenn in der Befehlszeile keine Variable angegeben wird. Standardwerte können auf zwei Arten angegeben werden: bei jeder Bezugnahme auf einen Parameter oder unter Verwendung des Befehls .DEF (für Default = Standardwert).

Wenn Sie den Standardwert bei jeder Bezugnahme auf einen Parameter angeben, müssen Sie die zwei Zeichenfolgen durch ein Dollarzeichen ($) trennen.

Beispiel: In der folgenden Anweisung:

```
ECHO "<wort1$defwort1> ist der Standard für Wort1."
```

ist "defwort1" der für "wort1" definierte Standardwert. Er wird ausgegeben, wenn für "wort1" keine andere Variable angegeben wird. Wenn Sie allerdings diesen Standardwert mehrmals in Ihrem Skript verwenden wollen, müssen Sie <wort1$defwort1> jedesmal wieder neu angeben.

Das Definieren eines Standardwerts mit dem Befehl .DEF ermöglicht die Angabe eines Standardwerts für jedes spezifische Schlüsselwort. Beispiel:

```
.DEF wort1 "defwort1"
```

ordnet "defwort1" als Standardwert für den Parameter "wort1" für das gesamte Skript zu. Die folgende Anweisung:

```
ECHO "<wort1> ist der Standardwert für Wort1."
```

führt zur gleichen Ausgabe wie der vorherige ECHO-Befehl:

```
defwort1 ist der Standardwert für Wort1.
```

Der Befehl .DOLLAR <z> ermöglicht das Ändern des Standardzeichens von $ zu <z>. (Sie können auch .DOL <z> angeben.) Beispiel:

```
.DOL #
```

```
ECHO "<wort1#defwort1> ist der Standardwert für "
ECHO "Wort1."
```

5.3.3 Kommentare

Sie können Kommentare in ein Skript einfügen, indem Sie sie hinter ein Semikolon (;) schreiben oder hinter einen Punkt (.), gefolgt von einem Leerzeichen. Leerzeilen werden innerhalb von Skripts akzeptiert und ignoriert.

Hinweis Es wird empfohlen, Semikolons zur Eingabe von Kommentaren zu verwenden, um Fehler zu vermeiden, die durch das Weglassen des Leerzeichens nach dem Punkt beim anderen Verfahren entstehen könnten.

5.3.4 Verschachteln von Befehlen

AmigaDOS bietet eine Reihe von Blockbefehlen, die in Skripts verwendet werden können, wie z. B. IF, ELSE, SKIP, LAB und QUIT. Diese Befehle können ebenso wie der Befehl EXECUTE in einem Skript auch verschachtelt werden.

Zum Stoppen der Ausführung des aktuellen Skripts drücken Sie Ctrl-D. Wenn Sie Skript-Dateien verschachtelt haben, können Sie die gesamte Gruppe von EXECUTE-Befehlen mit Ctrl-C stoppen.

Beispiel 1:

Angenommen, das Skript "Drucken" enthält folgende Angaben:

```
.KEY Dateiname
RUN COPY <Dateiname> TO PRT: +
ECHO "Drucken von <Dateiname> abgeschlossen"
```

Der folgende Befehl:

```
1> EXECUTE Drucken Test/Prg
```

reagiert so, als hätten Sie die folgenden Befehle über die Tastatur eingegeben:

```
1> RUN COPY Test/Prg TO PRT: +
ECHO "Drucken von Test/Prg abgeschlossen"
```

Beachten Sie die Verwendung des Pluszeichens am Ende der ersten Zeile. Wenn Sie nach dem Pluszeichen die Eingabetaste drücken, führt der Befehl RUN auch den Befehl in der zweiten Zeile aus, sobald der erste Befehl abgeschlossen ist.

Beispiel 2:

Ein weiteres Beispiel, Show, verwendet weitere der oben beschriebenen Funktionen:

```
.KEY name/A
IF EXISTS <name>
 TYPE <name> NUMBER   ;wenn Datei in angegebenem
                      ;Verzeichnis steht,
                      ;mit Zeilenzahlen anzeigen
ELSE
 ECHO "<name> steht nicht in diesem Verzeichnis"
ENDIF
```

Der Befehl:

```
1> EXECUTE Show Work/Docdatei
```

zeigt die Datei Work/Docdatei an, und zwar mit Zeilennummern, wenn "Show" im aktuellen Verzeichnis existiert. Wenn die Datei nicht dort vorhanden ist, wird eine Fehlermeldung am Bildschirm ausgegeben. Die Option /A erfordert die Angabe eines Dateinamens in der Befehlszeile nach "Show"; andernfalls kommt es zu einem Fehler.

5.3.5 Interaktive Skript-Dateien

Sie können auch Skripts erstellen, die vorübergehend anhalten, um Informationen vom Benutzer anzufordern, und danach die Verarbeitung fortsetzen. Die Befehle REQUESTCHOICE oder REQUESTFILE zum Verwenden der standardmäßigen Amiga-Dialogfenster bieten Ihnen die Möglichkeit, Benutzeraktionen im bekannten Workbench-Verfahren abzurufen. Dabei sind variable Bedingungen innerhalb eines einzelnen Skripts möglich. Beispiel: Das folgende Skript kopiert sechs Dateien von einer Festplatte auf eine Diskette. Wenn die sechs Dateien kopiert sind, fordert das Skript eine Bestätigung zum Fortsetzen des Kopiervorgangs an, was dem Benutzer Zeit gibt, bei Bedarf eine neue Diskette in das Laufwerk einzulegen.

```
COPY 2k.eps DF0:
COPY 2m.eps DF0:
COPY 2n.eps DF0:
COPY 2o.eps DF0:
COPY 2t.eps DF0:
COPY 2v.eps DF0:
ECHO "Dateien von Kapitel 2 kopiert."
ASK "Kopieren fortsetzen?"
IF WARN
   COPY 3a.eps DF0:
   COPY 3c.eps DF0:
   COPY 3g.eps DF0:
   COPY 3aa.eps DF0:
   COPY 3bb.eps DF0:
   COPY 3ff.eps DF0:
ENDIF
```

Bei der Eingabeaufforderung "Kopieren fortsetzen?" mit Y (Yes - Ja) antworten, um die restlichen Dateien auf eine (ggf. neue) Diskette im Laufwerk DF0: zu kopieren, oder mit N (No - Nein) den Kopiervorgang abbrechen.

5.3.6 Wiederholen von Befehlen

Sie können Skripts erstellen, die ein und denselben Befehl mehrmals wiederholen, wobei jeweils ein anderer Dateiname in die Befehlszeile eingesetzt wird. Beispiel: Zum Umbenennen von acht Dateien auf einmal können Sie folgendes Skript erstellen:

```
RENAME Abschn1 Kap1.1
RENAME Abschn2 Kap1.2
RENAME Abschn3 Kap1.3
RENAME Abschn4 Kap1.4
RENAME Abschn5 Kap1.5
RENAME Abschn6 Kap1.6
RENAME Abschn7 Kap1.7
RENAME Abschn8 Kap1.8
```

Bei diesem Beispiel wird davon ausgegangen, daß die Dateien im aktuellen Verzeichnis der Shell stehen. Wenn dies nicht der Fall ist, muß der vollständige Pfad zu jeder Datei angegeben werden.

Skripts wie dieses lassen sich häufig auch automatisch generieren, und zwar mittels der Option LFORMAT des Befehls LIST.

5.3.7 Beenden eines Skripts

Im allgemeinen enden Skripts, wenn alle darin angegebenen Befehle ausgeführt wurden. Rückkehrcodes melden, ob Befehle erfolgreich ausgeführt wurden oder gescheitert sind. Sie können auch einen Befehl QUIT in ein Skript schreiben, um es vorzeitig zu beenden, wenn es aber keine bestimmte Bedingung gibt, unter der Sie die Beendigung des Programms wünschen, ist ein QUIT-Befehl in einem Skript nicht nötig. Sie können ein Skript durch Drücken von Ctrl-D in dem Fenster abbrechen, in dem das Skript ausgeführt wird.

5.4 Bedingungskennzeichen

Bedingungskennzeichen legen die Bedingung fest, bei der die Ausführung eines bestimmten Befehls gestoppt wird. Wenn ein Befehl ausgeführt ist, gibt der Rückgabewert an, ob der Befehl erfolgreich oder erfolglos ausgeführt wurde. Es folgen die standardmäßigen Rückgabewerte:

0 Erfolgreiche Ausführung

5 Warnung. Ein Fehler ist aufgetreten, der jedoch nicht so schwerwiegend war, daß der Befehl hätte abgebrochen werden müssen. Wenn der Befehl in eine Befehlsdatei integriert ist, werden die nachfolgenden Befehle ausgeführt. Einige Befehle stellen das Bedingungskennzeichen auf WARN ein, um ein bestimmtes Ergebnis eines Befehls anzugeben, das nichts mit einem Fehler zu tun haben muß.

10 Fehler. Der Rückgabewert 10 bricht eine Befehlsdatei ab, wenn nicht mit dem Befehl FAILAT ein höherer Grenzwert angegeben wurde.

20 Erfolglose Ausführung.

Bei Anwendungsprogrammen werden eventuell andere Werte zurückgegeben. In diesem Fall werden die oben aufgelisteten Werte als untere Grenzwerte der jeweiligen Bedingung verwendet. Die Werte haben dann folgende Bedeutung:

0-4	Kein Fehler
5-9	Warnung
10-19	Fehler
20 oder höher	Erfolglose Ausführung

Manche Befehle, z. B. ASK und SEARCH, verwenden das Kennzeichen WARN, um bestimmte Situationen zu kennzeichnen, die in Skripts geprüft werden können.

Beispiel: Im COPY-Skript auf Seite 5-14 fordert der Befehl ASK eine Bestätigung zum Fortsetzen des Kopiervorgangs an:

```
ASK "Kopieren fortsetzen?"
IF WARN
   COPY 3a.eps DF0:
```

Wenn Sie Y eingeben, wird das Bedingungskennzeichen durch ASK auf 5 (WARN) gesetzt und somit der IF-Block ausgeführt. Wenn Sie aber N oder die Eingabetaste drücken, wird das Bedingungskennzeichen auf 0 (NO ERROR) gesetzt und die Befehlsdatei abgebrochen, weil die Anweisung IF nicht den entsprechenden Rückgabewert erhalten hat.

5.5 Fehlersuche bei Skript-Dateien

Wenn ein Skript-Befehl scheitert, sehen Sie möglicherweise eine der folgenden Fehlermeldungen:

```
<befehlsname>: Unbekannter Befehl
```

Sie haben etwas eingegeben, das nicht als Befehl erkannt wurde.

```
<befehlsname> fehlgeschlagen Rückgabewert 20
```

Die Argumente eines gültigen Befehls wurden inkorrekt eingegeben. Wenn diese Meldung erscheint, geben Sie hinter der Eingabeaufforderung WHY ein, um nähere Informationen über den Fehler zu erhalten.

Wenn ein Fehler auftritt, verwenden Sie Ihren Texteditor (ED, MEmacs oder ein anderes Textverarbeitungssystem, das ASCII-Dateien speichern kann), um die Zeile mit dem betreffenden Befehl zu korrigieren.

Das Einfügen der Zeile **SET ECHO ON** in das Skript ist bei der Lokalisierung von Fehlern von Nutzen. Diese Zeile bewirkt, daß alle darauf folgenden Zeilen als ECHO am Bildschirm angezeigt werden, bevor sie ausgeführt werden. Fehlermeldungen werden ausgegeben, nachdem das System versucht hat, den inkorrekten Befehl auszuführen. Zum Inaktivieren von SET ECHO geben Sie SET ECHO OFF ein oder löschen Sie die Zeile SET ECHO ON.

5.6 Umgebungsvariablen

Umgebungsvariablen dienen in Skripts zur Aufnahme von Status- und Zeichenfolgeninformationen. Variablen können für lange und umständlich einzugebende Zeichenfolgen substituiert werden. Das Verändern des Werts der Variablen ist bequemer als das erneute Edieren des Skripts, wenn sich der Wert für die Zeichenfolge ändern soll.

Umgebungsvariablen sind Variablen, die von AmigaDOS selbst verwaltet werden und nicht von Anwendungsprogrammen. Dies hat den Vorteil, daß auf diese Variablen von verschiedenen Programmen oder Befehlsdateien aus zugegriffen werden kann. Wenn in einer Befehlsdatei ein Dollarzeichen ($) mit einem Variablennamen dahinter angetroffen wird, wird diese Variable durch den ihr zugeordneten Wert ersetzt. Die Befehlszeile wird dann ausgeführt, als ob der Wert für die Variable eingegeben worden wäre.

AmigaDOS unterstützt beispielsweise die Variablen Workbench und Kickstart, die die aktuelle Versionsnummer von Workbench bzw. Kickstart beinhalten. Durch die folgende Befehlszeile wird die Workbench-Versionsnummer am Bildschirm ausgegeben:

```
ECHO "Amiga Workbench Version $Workbench"
```

Manche Variablen, z. B. die Variablen Workbench und Kickstart, wurden bereits erstellt. Die Shell berücksichtigt die Variable ECHO und aktualisiert die lokalen Variablen PROCESS, RC und RESULT2 automatisch. Die Variablen werden im folgenden erläutert:

ECHO Legt fest, ob Befehle vor der Ausführung auf dem Bildschirm angezeigt werden (ON) oder nicht (OFF). OFF ist der Standardwert.

PROCESS Prozeßnummer, wie <$$>.

RC Rückgabewert des zuletzt ausgeführten Befehls: 0, 5, 10 oder 20. Wird häufig in Befehlsdateien verwendet.

RESULT2 Sekundärer Rückgabewert bzw. Fehlernummer, die den Grund angibt, warum ein Befehl nicht ausgeführt wurde.

Wenn Sie beispielsweise am Anfang eines Skripts den Befehl SET ECHO ON eingeben, werden alle Befehlszeilen des Skripts auf dem Bildschirm angezeigt, bevor ihre Ausführung beginnt.

Wenn einer Umgebungsvariablen ein numerischer Wert zugeordnet wird, kann die Variable wie eine Zahl in Berechnungen und Ausdrücken verwendet werden. So können Sie z. B. der Variablen "neun" den Wert 9 zuordnen und dann $neun in EVAL-Ausdrücken verwenden. Beispiel:

```
1> SETENV neun 9
1> EVAL 5 * $neun
45
```

EVAL ist ein AmigaDOS-Befehl zur Berechnung ganzzahliger und Boolescher Ausdrücke. Der Befehl kann keine Umgebungsvariablen verarbeiten, die numerische Werte mit Hinterkommastellen aufweisen; achten Sie darauf, EVAL nur für ganzzahlige Werte zu verwenden.

5.6.1 Erstellen von Umgebungsvariablen

Umgebungsvariablen werden mit den Befehlen SET und SETENV erstellt.

5.6.1.1 SET

Mit SET können Sie lokale Variablen erstellen. Diese gelten nur für die Shell, in der sie erstellt wurden, und in allen aus dieser Original-Shell entstandenen untergeordneten Shells. Wenn Sie beispielsweise eine Umgebungsvariable im Shell-Fenster erzeugen und dann den Befehl NEWSHELL über den Menüpunkt "Befehl ausführen" ausführen, sind in dieser neuen Shell die in der ersten Shell erstellten Variablen nicht gültig. Hätten Sie allerdings eine zweite Shell geöffnet, indem Sie in der ersten Shell den Befehl NEWSHELL als Befehlszeile eingeben, würden alle Variablen der übergeordneten Shell auch für die neue Shell gelten.

Der Wert einer Variablen wird mit dem Befehl GET angezeigt, und mit dem Befehl UNSET wird eine Variable gelöscht.

5.6.1.2 SETENV

Mit SETENV werden globale Variablen erstellt, die für alle Shells gelten. Sie werden als kleine ASCII-Dateien im Verzeichnis ENV: gespeichert. Variablenwerte werden mit GETENV angezeigt, und mit UNSETENV werden globale Variablen gelöscht. Es ist empfehlenswert, globale Variablen nur dann einzusetzen, wenn auch andere Prozesse auf die Variablen zugreifen sollen.

Manche Anwendungen arbeiten mit Umgebungsvariablen. So unterstützt z. B. das Programm MORE eine Umgebungsvariable für den Editor. Sie können mit SETENV MEmacs als Ihren gewünschten Editor definieren:

```
1> SETENV Editor Extras:Tools/MEmacs
```

Achten Sie darauf, den vollständigen Pfad zu MEmacs anzugeben.

Wenn Sie mit MORE den Inhalt der Datei User-startup anzeigen, schaltet die Tastenkombination Shift-E dann automatisch in einen MEmacs-Bildschirm, in dem die Datei User-startup geladen ist und zur Bearbeitung bereitsteht.



Kapitel 6

AmigaDOS-Befehlsreferenz

Die in diesem Kapitel beschriebenen Befehle werden von der Shell aus aufgerufen. Bis auf einige Befehle am Ende des Kapitels, die für die Verwendung durch das System reserviert sind, erfolgen die Erläuterungen in alphabetischer Reihenfolge.

Die folgende Tabelle enthält kurze, alphabetisch sortierte Referenzangaben zu den in diesem Kapitel beschriebenen Befehlen, deren Funktion und der Seite, auf der der jeweilige Befehl erläutert wird:

Befehl	Funktion	Seite
ADDBUFFERS	Dateisystem anweisen, Cache-Puffer eines Laufwerks anzuzeigen bzw. hinzuzufügen.	6-12
ADDDATATYPES	Eine Liste der Datentypen aufbauen (Systembefehl).	6-109
ALIAS	Alias-Namen für Befehle definieren und anzeigen.	6-13
ASK	Angeben, ob während der Ausführung einer Befehlsdatei eine Eingabeaufforderung angezeigt werden soll.	6-14
ASSIGN	Zuweisung logischer Geräte steuern.	6-15
AVAIL	Verfügbare Speicherkapazität anzeigen.	6-21
BINDDRIVERS	Gerätetreiber in der Schublade "Expansion" aktivieren (Systembefehl).	6-110
BREAK	Unterbrechungskennungen im angegebenen Prozeß aktivieren.	6-22

Befehl (Forts.)	Funktion (Forts.)	Seite (Forts.)
CD	Aktuelles Verzeichnis wechseln oder anzeigen.	6-23
CHANGETASKPRI	Priorität von Shell-Prozessen ändern.	6-25
CONCLIP	Daten vom Konsolenfenster zum Zwischenspeicher und umgekehrt übertragen (Systembefehl).	6-110
COPY	Dateien oder Verzeichnisse kopieren.	6-26
CPU	Prozessoroptionen einstellen oder anzeigen.	6-29
DATE	Systemdatum und -zeit einstellen oder anzeigen.	6-31
DELETE	Dateien und Verzeichnisse löschen.	6-33
DIR	Dateien eines Verzeichnisses in einer sortierten Liste anzeigen.	6-35
DISKCHANGE	Amiga informieren, daß in ein Laufwerk eine andere Disk eingelegt wurde.	6-38
ECHO	Zeichenkette anzeigen.	6-39
ED	Textdateien edieren.	6-39
EDIT	Textdateien edieren (Zeileneditor).	6-40
ELSE	Alternative für eine IF-Anweisung in einer Befehlsdatei angeben.	6-40
ENDCLI	Shell-Prozeß beenden.	6-41
ENDIF	IF-Block in einer Befehlsdatei beenden.	6-42
ENDSHELL	Shell-Prozeß beenden.	6-42
ENDSKIP	SKIP-Block in einer Befehlsdatei beenden.	6-43
EVAL	Ganzzahligen oder Booleschen Ausdruck auswerten.	6-43
EXECUTE	Befehlsdatei aufrufen (eventuell mit variablen Argumenten)	6-45
FAILAT	Angeben, daß eine Befehlsfolge erst abgebrochen wird, wenn ein bestimmter Rückgabecode gesendet wird.	6-46

Befehl (Forts.)	Funktion (Forts.)	Seite (Forts.)
MOUNT	Ein an das System angeschlossenes Gerät verfügbar machen.	6-68
NEWCLI	Neues Shell-Fenster öffnen.	6-70
NEWSHELL	Neues Shell-Fenster öffnen.	6-70
PATH	Liste der Verzeichnisse verwalten, in denen die Shell Befehle sucht.	6-74
PROMPT	Eingabeaufforderungstext der aktuellen Shell ändern.	6-76
PROTECT	Schutzbits einer Datei oder eines Verzeichnisses ändern.	6-77
QUIT	Befehlsdatei mit einem angegebenen Rückgabecode verlassen.	6-79
RELABEL	Datenträgernamen der Disk im angegebenen Laufwerk auf den angegebenen Namen ändern.	6-80
REMRAD	Resetfeste RAM-Disk RAD: entfernen.	6-81
RENAME	Namen bzw. Standort einer Datei oder eines Verzeichnisses ändern.	6-81
REQUESTCHOICE	Ermöglichen, daß AmigaDOS- und ARexx-Befehlsdateien benutzerspezifische Dialogfenster verwenden.	6-82
REQUESTFILE	Ermöglichen, daß AmigaDOS- und ARexx-Befehlsdateien Dateiauswahlfenster verwenden.	6-84
RESIDENT	Liste residenter Befehle anzeigen und bearbeiten.	6-86
RUN	Befehle als Hintergrundprozesse ausführen.	6-89
SEARCH	Angegebene Zeichenkette in den Dateien der angegebenen Verzeichnisse suchen.	6-90
SET	Lokale Variable definieren.	6-92
SETCLOCK	Batteriegepufferte Echtzeituhr stellen bzw. Zeit abfragen.	6-93

6.1 Befehlsbeschreibung

Jeder in diesem Handbuch dokumentierte Befehl wird mit folgenden Angaben zur ordnungsgemäßen Verwendung des Befehls beschrieben: Format, Argumente, Optionen, Symbole und Abkürzungen.

Dieses Kapitel und Kapitel 7 enthalten Befehlsspezifikationen für die AmigaDOS-Befehle und die Workbench-Programme, auf die über die Shell zugegriffen werden kann. Jeder Befehl wird nach folgendem Schema erläutert:

Format Alle Argumente und Optionen, die von einem Befehl akzeptiert werden. Die Sonderzeichen zur Angabe eines bestimmten Argumenttyps werden in Kapitel 6.1.1 beschrieben.

Schablone Wahlfreie Online-Erinnerung an das Befehlsformat, die in den Programmcode eingebettet ist. Wenn Sie einen Befehl gefolgt von einem Leerzeichen und einem Fragezeichen eingeben (z. B. `DIR ?`), wird die zugehörige Schablone angezeigt. In Kapitel 6.1.2 wird die Schablone detailliert beschrieben.

Pfad Verzeichnis, in dem der Befehl normalerweise gespeichert ist.

Beispiele Beispielverwendungen des Befehls. Zur besseren Unterscheidung vom normalen Text werden die Beispiele in der Schriftart "Courier" dargestellt. Die Zeichenkette `1>` steht für die Shell-Eingabeaufforderung. Geben Sie sie nicht als Teil des Beispielbefehls ein. Zeilen im Beispiel, denen die Zeichenkette `1>` nicht vorangestellt ist, zeigen die Ausgabedaten des jeweiligen Befehls. Befehlsnamen und Schlüsselwörter sind aus Gründen der Übersichtlichkeit nur mit Großbuchstaben geschrieben. Datei- und Verzeichnisnamen beginnen in der Regel mit einem Großbuchstaben. Bei der Eingabe muß jedoch die Groß- und Kleinschreibung nicht berücksichtigt werden. Die Befehle werden ausgeführt, sobald Sie die Eingabetaste drücken.

Befehle und Argumente müssen durch mindestens ein Leerzeichen voneinander getrennt werden. Andere Satzzeichen dürfen nur dann verwendet werden, wenn es die Syntax des jeweiligen Befehls erfordert.

6.1.1 Format

In der folgenden Liste sind die Sonderzeichen aufgeführt, die in Formatbeschreibungen zur Angabe des Argumenttyps dienen. Geben Sie diese Zeichen nicht als Teil des Befehls ein.

< >
: Argumente, die unbedingt als zusätzliche Information (z. B. ein Dateiname) angegeben werden müssen, stehen in spitzen Klammern. Argumente in spitzen Klammern müssen nur dann nicht eingegeben werden, wenn sie wiederum von eckigen Klammern umschlossen werden (Beispiel: [<Dateiname>]; siehe unten).

[ ]
: Wahlfreie Argumente und Schlüsselwörter stehen in eckigen Klammern. Diese werden zwar von dem Befehl akzeptiert, sind aber nicht erforderlich.

{ }
: Elemente, die einmal oder beliebig oft angegeben werden können stehen in geschweiften Klammern. {<Argumente>} bedeutet beispielsweise, daß für dieses Argument mehrere Elemente angegeben werden können.

|
: Senkrechte Striche bedeuten "oder" und dienen zum Trennen von Optionen, von denen Sie jeweils nur eine auswählen können. [OPT R|S|RS] bedeutet z. B., daß Sie entweder die Option R, die Option S oder beide Optionen, also RS, angeben können.

<n>
: Als Argument wird ein numerischer Wert erwartet.

KEYWORD
: Kursivschrift gibt an, daß bei Angabe des Arguments das zugehörige Schlüsselwort verwendet werden muß.

...
: Eine Ellipse (...) hinter einem Zeichenkettenargument bedeutet, daß die Zeichenkette als letztes in der Befehlszeile stehen muß. Kommentare sind in diesem Fall nicht zulässig. Der Rest der Befehlszeile wird als die gewünschte Zeichenkette interpretiert. Selbst wenn die Zeichenkette Leerzeichen enthält, muß die Zeichenkette nicht in Anführungszeichen gesetzt werden. Wenn Sie Anführungszeichen angeben, werden sie als Teil der Zeichenkette interpretiert. Wenn Sie das Schlüsselwort angeben, können Sie der Zeichenkette Leerzeichen voran- und nachstellen.

Einrückung der Befehlszeile Bei Befehlszeilen, die aufgrund ihrer Länge einen Zeilenumbruch enthalten, werden die folgenden Zeilen nur zu Dokumentationszwecken eingerückt angezeigt. In der Praxis werden die umgebrochenen Zeilen am ersten Zeichen der Shell-Eingabeaufforderung ausgerichtet.

Im folgenden wird am Befehl COPY die Verwendung dieser Konventionen erläutert:

```
COPY[FROM]{<Name|Namensmuster>}
    [TO]<Name|Namensmuster>[ALL]
    [QUIET][BUF/BUFFER=<n>][CLONE][DATES][NOPRO]
    [COM][NOREQ]
```

[FROM] ist ein wahlfreies Schlüsselwort. Wenn es nicht angegeben wird, findet der Befehl den zu kopierenden Dateinamen oder das Muster aufgrund der Position in der Befehlszeile.

<Name|Namensmuster> ist ein obligatorisches Argument. Sie müssen einen Dateinamen oder ein Namensmuster eingeben. Durch die geschweiften Klammern wird angezeigt, daß mehr als ein Argument eingegeben werden kann.

[TO] ist ein wahlfreies Schlüsselwort. Wenn es nicht angegeben wird, findet der Befehl den Namen der Zieldatei oder des Zielgeräts aufgrund der Position in der Befehlszeile.

<Name|Namensmuster> ist ein obligatorisches Argument. Es kann nur ein Ziel angegeben werden.

[ALL], [QUIET], [CLONE], [DATES], [NOPRO], [COM] und [NOREQ] sind wahlfreie Argumente.

[*BUF|BUFFER*=<n>] ist ein wahlfreies Argument. Wenn es angegeben wird, können Sie BUF oder BUFFER in Verbindung mit dem numerischen Argument eingeben. So sind z. B. BUF=5 und BUFFER=5 gültige Argumente. Das numerische Argument kann auch ohne Gleichheitszeichen eingegeben werden, dann müssen Sie Leerzeichen als Trennzeichen verwenden.

6.1.2 Schablone (engl. Template)

Die Schablone ist in das System integriert und dient als Online-Erinnerung für die Befehlssyntax. Außerdem können Sie einen Befehl von der Schablonenzeile aus aufrufen, indem Sie hinter der angezeigten Eingabeaufforderung die Argumente des gewünschten Befehls eingeben.

Die Befehlsschablone wird angezeigt, wenn Sie hinter dem gewünschten Befehl ein Fragezeichen (?) eingeben. Die Shell geht davon aus, daß der Befehl aufgerufen werden soll. Vom Benutzer wird erwartet, daß er die Argumente des Befehls hinter dem Doppelpunkt eingibt, der der angezeigten Schablone nachgestellt ist. Beispiel:

```
1> TYPE ?
FROM/A/M,TO/K,OPT/K,HEX/S,NUMBER/S:
```

Sind zur ordnungsgemäßen Ausführung eines Befehls keine Argumente erforderlich, wird er nach Drücken der Eingabetaste ausgeführt. Wenn Sie die erforderlichen Argumente und die zugehörigen Schlüsselwörter eingeben und anschließend die Eingabetaste drücken, werden Befehle ebenfalls ausgeführt. Wenn Sie einen Befehl ohne seine erforderlichen Argumente oder mit falschen Argumenten eingeben und anschließend die Eingabetaste drücken, erscheint eine entsprechende Fehlermeldung, die jedoch keinen schwerwiegenden Systemfehler bedeutet. Beachten Sie, daß Sie hinter dieser Eingabeaufforderung nicht das vollständige Befehlsformat eingeben müssen, sondern die erforderlichen Argumente ausreichend sind.

In Befehlsschablonen werden Argumente durch Kommas getrennt. Hinter dem Argument stehen ein Schrägstrich (/) und ein Großbuchstabe, der die Art des Arguments angibt. Diese Kombination aus Schrägstrich und Kennbuchstabe, die nicht als Teil des Befehls eingegeben werden dürfen, sollen Sie an die spezifischen Erfordernisse für ein Argument erinnern. Ein Argument kann mehrere solcher Kennzeichen aufweisen. Die Bedeutung der einzelnen Kombinationen wird in der folgenden Tabelle erläutert:

Schablonenargument	Formatentsprechung	Bedeutung
Argument/A	<Name>	Dieses Argument muß immer angegeben werden.
Option/K	*KEYWORD*	Das Schlüsselwort der Option ist erforderlich, wenn das Argument angegeben wird. (Die Schlüsselwörter dürfen immer mit angegeben werden.)
Option/S	[KEYWORD]	Diese Option fungiert als Schalter. Sie müssen den Namen dieser Option eingeben, um diese Option auszuwählen. Die meisten Optionen sind Schalter.
Wert/N	<n>	Das Argument ist ein numerischer Wert.
Argument/M	{<Name>}	Für dieses Argument werden mehrere Elemente akzeptiert. Die Zahl der Elemente unterliegt keinen Beschränkungen. Mehrfachargumente müssen jedoch vor anderen Argumenten und Optionen eingegeben werden.
Zeichkette/F	argument...	Die Zeichenkette muß das abschließende Argument der Befehlszeile sein. Der Rest der Befehlszeile wird als die gewünschte Zeichenkette verarbeitet.

Schablonen-argument (Forts.)	Formatentsprechung (Forts.)	Bedeutung (Forts.)
=	KYWD | KEYWORD	Ein Gleichheitszeichen zeigt an, daß zwei verschiedene Formen des Schlüsselworts die gleiche Bedeutung haben. Beide Formen sind gleichermaßen gültig. Das Gleichheitszeichen ist nicht Teil des Befehls und darf nicht eingegeben werden.

Im folgenden wird am Beispiel der Schablone des Befehls COPY die Verwendung von Argumenten erläutert:

```
FROM/M,TO/A,ALL/S,QUIET/S,BUF=BUFFER/K/N,
   CLONE/S,DATES/S,NOPRO/S,COM/S,NOREQ/S
```

FROM/M zeigt an, daß ein Argument angegeben werden muß und daß mehrere Argumente möglich sind.

TO/A zeigt an, daß das Argument angegeben werden muß.

ALL/S, QUIET/S, CLONE/S, DATES/S, NOPRO/S und COM/S zeigen an, daß diese Schlüsselwörter als Schalter fungieren. Bei Angabe des Schlüsselworts in der Befehlszeile wird diese Option verwendet.

BUF=BUFFER/K/N zeigt an, daß bei Verwendung von BUF oder BUFFER das Schlüsselwort (/K) und ein numerisches Argument (/N) angegeben werden muß. Dabei sind BUF und BUFFER gültige Schlüsselwörter (=).

Allgemein können Schlüsselwörter und zugehörige Argumente durch Gleichheitszeichen (=) verbunden werden, um in komplexeren Befehlszeilen eine eindeutige Zuordnung sicherzustellen (z. B. BUF=20).

6.2 Befehlsliste

ADDBUFFERS

Mit diesem Befehl wird das Dateisystem angewiesen, für ein Laufwerk Cache-Puffer hinzuzufügen bzw. die verfügbaren Cache-Puffer anzuzeigen.

Format ADDBUFFERS <Laufwerk> [<n>]

Schablone DRIVE/A,BUFFERS/N

Pfad C:

Mit dem Befehl ADDBUFFERS wird die Anzahl der für ein <Laufwerk> verfügbaren Puffer um <n> erhöht. Durch Hinzufügen weiterer Puffer wird zwar der Zugriff auf die Disk erheblich beschleunigt, aber jeder zusätzliche Puffer verringert die Größe des verfügbaren Speichers um ca. 512 Byte. Standardmäßig werden Diskettenlaufwerken 5 Puffer zugewiesen und Festplatten 30 Puffer.

Die Anzahl der Puffer ist immer auf die Kapazität des insgesamt verfügbaren Speicherplatzes abzustimmen. Eine feste Obergrenze gibt es nicht, aber wenn zu viele Puffer zugewiesen werden, wird die Gesamtleistung reduziert, da sie den für andere Funktionen benötigten RAM-Speicherplatz belegen. Wenn Sie eine negative Zahl eingeben, werden entsprechend viele Pufferzuweisungen aufgehoben. Es muß mindestens ein Puffer zugewiesen werden. Dieser Mindestwert ist jedoch nicht empfehlenswert.

In einem 512-KB-System werden für ein Diskettenlaufwerk 20 Puffer empfohlen. Für Festplatten verwenden Sie den durch das Programm "HDToolbox" empfohlenen Vorgabewert (dieser Wert kann durch Auswahl von "Advanced Options" auf dem Bildschirm "Partitioning" angezeigt werden).

Wenn Sie nur das Argument <Laufwerk> angegeben haben, wird die aktuelle Anzahl der Puffer für das jeweilige Laufwerk angezeigt.

Beispiel:

```
1> ADDBUFFERS DF0:
DF0: has 5 buffers
```

Kapitel 8 enthält ein weiteres Beispiel zur Verwendung des Befehls ADDBUFFERS.

ALIAS

Mit diesem Befehl können Sie Alias-Namen für Befehle definieren oder anzeigen.

Format ALIAS [<Name>] [<Zeichenkette...>]

Schablone NAME,STRING/F

Pfad Intern

Mit dem Befehl ALIAS werden Alias-Namen bzw. alternative Namen für AmigaDOS-Befehle definiert. Dadurch können Sie häufig verwendete Befehle abkürzen oder Standardbefehle umbenennen.

Bei Angabe von <Name> in der Befehlszeile wird der Name durch die definierte <Zeichenkette> ersetzt. Diese wird in die Befehlszeile integriert, die dann wiederum in dieser Form von AmigaDOS interpretiert und ausgeführt wird. <Name> ist der Alias-Name für den Befehl, und <Zeichenkette> ist der Befehl, der für den Alias-Namen eingesetzt wird.

Ein Alias-Name muß am Anfang der Befehlszeile eingegeben werden. Nach dem Alias-Namen können Argumente angegeben werden. Es kann jedoch kein Alias-Name erstellt werden, der nur für eine Folge von Befehlsargumenten steht. Beispiel:

```
1> NEWSHELL WINDOW=CON:0/250/640/150/2SHELL/CLOSE
```

In dieser Befehlszeile kann beispielsweise das Argument WINDOW nicht durch einen Alias-Namen ersetzt werden.

Sie können einen Dateinamen oder andere Befehlsteile eines Alias-Namens ersetzen, wenn Sie dazu in der <Zeichenkette> eckige Klammern ([]) ohne dazwischenliegende Zeichen verwenden. Hinter dem Alias-Namen eingegebene Argumente werden später an der Klammerposition eingesetzt.

Mit dem verkürzten Befehl ALIAS <Name> wird die zugehörige <Zeichenkette> angezeigt. Wenn Sie nur ALIAS eingeben, werden alle aktuellen Alias-Namen angezeigt.

Wenn Sie mit dem Befehl NEWSHELL eine neue Shell erstellen, können die Alias-Namen gemeinsam mit der übergeordneten Shell verwendet werden. Wenn Sie jedoch über den Menüpunkt "Befehl ausführen" eine neue Shell öffnen, erkennt diese Shell die Alias-Namen einer anderen Shell nicht. Wenn Sie einen globalen Alias-Namen erstellen wollen, der in allen Shells gültig sein soll, fügen Sie den entsprechenden ALIAS-Befehl in die Datei "S:Shell-startup" ein.

Mit dem Befehl UNALIAS können Sie Alias-Namen löschen.

Beispiel 1:

```
1> ALIAS d1 DIR DF1:
```

Wenn Sie "d1" eingeben, wird das Inhaltsverzeichnis der Diskette in Laufwerk DF1: angezeigt. Dies entspricht der Eingabe von "DIR DF1:".

Beispiel 2:

```
1> ALIAS hex TYPE [] HEX
```

Der Alias-Name "hex" wird erstellt. Damit wird der Inhalt der angegebenen Datei in Hexadezimalformat angezeigt. Die eckigen Klammern zeigen an, wo der Dateiname einzufügen ist. Beispiel:

```
1> hex MeineDatei
```

Mit dieser Eingabe wird der Inhalt von "MeineDatei" in Hexadezimalformat angezeigt.

Siehe auch: UNALIAS. Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls ALIAS.

ASK

Mit diesem Befehl wird während der Ausführung einer Befehlsdatei eine Ja-/Nein-Entscheidung vom Benutzer erfragt.

Format	ASK <Eingabeaufforderung>
Schablone	PROMPT/A
Pfad	Intern

Der Befehl ASK wird in Befehlsdateien verwendet, um den mit <Eingabeaufforderung> angegebenen Text im aktuellen Fenster anzuzeigen und auf eine Tastatureingabe zu warten. Sie können "Y" (Yes=Ja) oder "N" (No=Nein) eingeben oder nur die Eingabetaste drücken (gleichbedeutend mit Nein). Bei Eingabe von "Y" wird das Bedingungskennzeichen auf 5 (WARN) gesetzt. Wenn Sie "N" eingeben oder Eingabetaste drücken, wird es auf 0 gesetzt. Sie können dies anschließend mit dem Befehl IF WARN überprüfen.

Wenn die <Eingabeaufforderung> Leerzeichen enthält, muß sie zwischen Anführungszeichen stehen.

Beispiel:

Es liegt eine Befehlsdatei mit folgenden Befehlen vor:

```
ASK Weiter?
IF WARN
   ECHO Ja
ELSE
   ECHO Nein
ENDIF
```

Wenn der Befehl ASK abgearbeitet wird, erscheint die Eingabeaufforderung **`Weiter?`**. Wenn Sie "Y" eingeben (und anschließend die Eingabetaste drücken), wird **`Ja`** angezeigt. Wenn Sie nur die Eingabetaste drücken oder "N" eingeben, erscheint **`Nein`**.

Siehe auch: IF, ELSE, ENDIF, REQUESTCHOICE, WARN

ASSIGN

Mit diesem Befehl werden Dateien und Verzeichnissen logische Gerätenamen zugewiesen.

Format	ASSIGN [<Name>:] [{<Ziel>}] [LIST] [EXISTS] [DISMOUNT] [DEFER] [PATH] [ADD] [REMOVE] [VOLS] [DIRS] [DEVICES]
Schablone	NAME,TARGET/M,LIST/S,EXISTS/S, DISMOUNT/S,DEFER/S,PATH/S,ADD/S, REMOVE/S,VOLS/S,DIRS/S,DEVICES/S
Pfad	C:

Durch den Befehl ASSIGN können Sie Dateien und Verzeichnisse anstatt über ihren normalen Namen oder den vollständigen Pfad über einen kürzeren, benutzerfreundlicheren logischen Gerätenamen aufrufen. Mit ASSIGN können Sie Zuweisungen erstellen und löschen sowie einige oder alle aktuellen Zuweisungen auflisten.

Wenn die Argumente <Name> und {<Ziel>} angegeben werden, erhält das angegebene Ziel den angegebenen logischen Gerätenamen. Bei jedem Bezug auf den zugewiesenen logischen Gerätenamen greift AmigaDOS auf das entsprechende Verzeichnis zu. Wenn der vergebene <Name> bereits einem anderen Verzeichnis zugewiesen war, wird das alte Ziel durch das neue ersetzt. Hinter dem Argument <Name> muß immer ein Doppelpunkt gesetzt werden.

Wenn nur das Argument <Name> angegeben wird, werden alle Zuweisungen eines Verzeichnisses bzw. einer Datei für dieses logische Gerät aufgehoben.

Sie können einem Verzeichnis mehrere logische Gerätenamen zuweisen, indem Sie den Befehl ASSIGN entsprechend oft aufrufen.

Sie können einem logischen Gerätenamen mehrere Ziele zuweisen, indem Sie hinter dem Argument <Name> alle gewünschten Dateien und Verzeichnisse angeben oder mehrere ASSIGN-Befehle mit der Option ADD aufrufen. Bei Angabe der Option ADD werden bestehende Zuweisungen zwischen <Name> und <Ziel> nicht aufgehoben, sondern das neu angegebene Ziel wird in der Zuweisungsliste hinzugefügt. Das System sucht dann nach beiden Verzeichnissen, wenn es auf <Name> trifft. Wenn das zuerst zugewiesene Verzeichnis nicht zur Verfügung steht, wird das neu zugewiesene verwendet.

Mit der Option REMOVE können Sie einen Zielnamen aus der Zuweisungsliste löschen.

Wenn Sie ASSIGN ohne Argumente eingeben oder das Schlüsselwort LIST verwenden, wird eine Liste aller aktuellen Zuweisungen angezeigt. Wenn die Schalter VOLS, DIRS oder DEVICES angegeben sind, wird die Anzeige auf Datenträger, Verzeichnisse bzw. Geräte beschränkt.

Wenn Sie das Schlüsselwort EXISTS mit einem logischen Gerätenamen angeben, wird die Zuweisungsliste nach diesem Namen durchsucht. Anschließend werden der Datenträger und das Ziel angezeigt, die diesem Gerät zugewiesen sind. Wird der Gerätename

nicht gefunden, wird das Bedingungskennzeichen auf 5 (WARN) gesetzt.

Wenn das Argument {<Ziel>} angegeben wird, sucht AmigaDOS sofort nach der Datei bzw. dem Verzeichnis. Wenn die ASSIGN-Befehle in die Startsequenz ("User-startup") integriert sind, müssen die Ziele während des Startvorgangs auf einer angemeldeten Disk vorhanden sein. Wenn ein zugewiesenes Ziel nicht gefunden werden kann, erscheint ein Dialogfenster, in dem Sie aufgefordert werden, den Datenträger einzulegen, der das Verzeichnis enthält. Bei den beiden Optionen DEFER und PATH wird jedoch mit der Suche gewartet, bis das Verzeichnis benötigt wird.

Hinweis Der zugewiesene Name muß nicht den Namen der Datei bzw. des Verzeichnisses beinhalten, und er muß nicht in Großbuchstaben geschrieben werden. Dem Verzeichnis "Ram Disk:Clipboards" können beispielsweise die Namen CLIPS: oder Clips: zugeordnet werden.

Mit der Option DEFER wird eine "nachträgliche" Zuweisung erstellt. Diese tritt nicht direkt bei der Zuweisung in Kraft, sondern erst wenn zum ersten Mal auf das Objekt Bezug genommen wird. Dadurch ist die Disk, auf der sich das angegebene Ziel befindet, erst erforderlich, wenn auf das Objekt zugegriffen wird. Anschließend bleibt die Zuweisung so lange unverändert gültig, bis sie ausdrücklich geändert wird.

Wenn Sie z. B. mit der Option DEFER dem Verzeichnis "DF0:Fonts" den Namen "FONTS:" zuweisen, wird FONTS: der Diskette zugeordnet, die sich gerade in Laufwerk DF0: befindet, wenn FONTS: später angesprochen wird. Wenn sich zu diesem Zeitpunkt gerade eine Workbench-Diskette in Laufwerk DF0: befindet, wird FONTS: dieser Diskette zugeordnet. Wenn Sie die Diskette später herausnehmen und dann eine andere Diskette einlegen, auf der sich ein Verzeichnis mit dem Namen "Fonts" befindet, werden Sie trotzdem aufgefordert, die ursprüngliche Workbench-Diskette wieder einzulegen, wenn FONTS: das nächste Mal angesprochen wird.

Mit der Option PATH wird dagegen eine "unverbindliche" Zuweisung erstellt. Sie entspricht in der Funktion etwa einer Zuweisung mit der

Option DEFER. Der Unterschied besteht jedoch darin, daß die Zuweisung bei jeder Bezugnahme auf den zugewiesenen Namen neu identifiziert wird. Wenn Sie z. B. mit der Option PATH dem Verzeichnis "DF0:Fonts" den Namen "FONTS:" zuweisen, sucht das System bei der Bezugnahme auf FONTS: auf derjenigen Disk, die sich gerade im Laufwerk befindet, auch wenn diese gewechselt wurde. Wenn diese Disk ein Verzeichnis mit dem Namen "Fonts" enthält, wird dieses für den Befehl ASSIGN verwendet. Mit der Option PATH können Sie jedoch nicht mehrere Verzeichnisse zuweisen.

Die Option PATH wird vor allem von Benutzern von Diskettensystemen verwendet, da sie nicht mehr die Original-Workbench-Diskette einlegen müssen, mit der das System gestartet wurde. Solange sich in dem Laufwerk, das Sie mit der Option PATH zugewiesen haben, irgendeine Diskette mit dem zugewiesenen Verzeichnis befindet, benutzt das System diese Diskette.

Mit der Option DISMOUNT wird ein Datenträger oder ein Gerät aus der Liste der angemeldeten Geräte gelöscht. In diesem Argument muß der Gerätename angegeben werden. Über DISMOUNT wird lediglich der Name aus der Liste entfernt, aber kein Speicherplatz freigegeben. Die einzige Möglichkeit, DISMOUNT rückgängig zu machen, ist ein Neustart. DISMOUNT ist ausschließlich für Softwareentwickler gedacht. Unsachgemäßer Umgang mit dieser Option kann zu Softwarefehlern führen.

Beispiel 1:

```
1> ASSIGN FONTS: MeineFonts:Fontdir
```

Mit diesem Befehl wird dem Verzeichnis "Fontdir" auf dem Datenträger "MeineFonts" der Name "FONTS:" zugewiesen.

Beispiel 2:

```
1> ASSIGN LIST
Volumes:
Ram Disk [Mounted]
Workbench [Mounted]
MeineFonts [Mounted]

Directories:
LOCALE      Workbench:Locale
KEYMAPS     Workbench:Devs/Keymaps
```

```
PRINTERS     Workbench:Devs/Printers
REXX         Workbench:S
CLIPS        Ram Disk:Clipboards
ENV          Ram Disk:Env
T            Ram Disk:T
ENVARC       Workbench:Prefs/Env-Archive
SYS          Workbench:
C            Workbench:C
S            Workbench:S
L            Workbench:L
FONTS        MyFonts:Fontdir
DEVS         Workbench:Devs
LIBS         Workbench:Libs
           + Workbench:Classes

Devices:
PIPE AUX RAM CON
RAW PAR SER PRT DF0
```

Mit diesem Befehl werden die aktuellen Zuweisungen (hier ein typisches Ergebnis) angezeigt. Das Pluszeichen gibt zusätzliche Verzeichnisse mit derselben Zuweisung an.

Beispiel 3:

```
1> ASSIGN FONTS: EXISTS
FONTS: MeineFonts:FontDir
```

Mit diesem Befehl werden die Zuweisungen von FONTS: abgefragt. Die Abfrage ergibt, daß FONTS: dem Verzeichnis "Fontdir" auf dem Datenträger "MeineFonts" zugewiesen ist. Ist der Gerätename vorhanden, wird der Rückgabecode auf 0 gesetzt, ansonsten auf 5.

Beispiel 4:

```
1> ASSIGN LIBS: SYS:Libs BigAssem:Libs ADD
```

Mit diesem Befehl werden mehrere Verzeichnisse zugewiesen. Diese bilden einen Suchpfad, der die zwei "Libs"-Verzeichnisse enthält. Durch die Angabe von ADD wird die Standardzuweisung "SYS:Classes" (siehe Beispiel 2) nicht entfernt. Diese Verzeichnisse werden bei jedem Zugriff auf LIBS: der Reihe nach durchsucht.

Beispiel 5:

```
1> ASSIGN DEVS:
```

Die Zuweisung DEVS: wird gelöscht.

Beispiel 6:

```
1> ASSIGN WorkDisk: DF0: DEFER
1> ASSIGN WorkDisk: EXISTS
WorkDisk <DF0:>
```

Mit diesem Befehl wird eine nachträgliche Zuweisung für das logische Gerät "WorkDisk:" erstellt. Diese Diskette muß erst dann in Laufwerk DF0: eingelegt werden, wenn Sie das erste Mal auf den Namen "WorkDisk:" Bezug nehmen. Beachten Sie, daß DF0: zunächst in spitzen Klammern angezeigt wird. Dies bedeutet, daß die Option DEFER aktiv ist. Nach der ersten Benutzung von "WorkDisk:" wird die vorläufige Angabe <DF0:> durch den Datenträgernamen der Diskette in Laufwerk DF0: ersetzt.

Beispiel 7:

```
1> ASSIGN C: DF0:C PATH
1> ASSIGN C: EXISTS
C   [DF0:C]
```

Mit diesem Befehl wird bei der Befehlssuche auf das Verzeichnis C: derjenigen Disk zugegriffen, die sich gerade in DF0: befindet. Beachten Sie, daß DF0:C in eckigen Klammer angezeigt wird. Dies bedeutet, daß es sich um eine unverbindliche Zuweisung handelt.

Beispiel 8:

```
1> ASSIGN LIBS: ZCad:Libs ADD
```

Mit diesem Befehl wird "ZCad:Libs" in die Liste der Verzeichnisse aufgenommen, die dem Namen "LIBS:" zugewiesen sind.

Beispiel 9:

```
1> ASSIGN LIBS: ZCad:Libs REMOVE
```

Mit diesem Befehl wird "ZCad:Libs" aus der Liste der Verzeichnisse gelöscht, die dem Namen "LIBS:" zugewiesen sind.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls ASSIGN.

AVAIL

Mit diesem Befehl wird der verfügbare Speicherplatz im Chip-RAM und im Fast-RAM angezeigt.

Format	AVAIL [CHIP|FAST|TOTAL] [FLUSH]
Schablone	CHIP/S,FAST/S,TOTAL/S,FLUSH/S
Pfad	C:

AVAIL zeigt einen Überblick über das System-RAM, Chip- und Fast-RAM. Für jeden Speichertyp werden die Gesamtkapazität (Maximum), der verfügbare (Available) und der belegte (In-Use) Speicherplatz sowie der größte zusammenhängende, nicht belegte Speicherblock (Largest) angezeigt.

Mit den Optionen CHIP, FAST und/oder TOTAL wird nur die Zahl der verfügbaren Bytes des Chip- oder des Fast-RAM oder beider zusammen angezeigt.

Mit der Option FLUSH wird Speicherplatz freigegeben, indem alle nicht benutzten Bibliotheken, Gerätetreiber und Schriftarten gelöscht werden.

Beispiel 1:

```
1> AVAIL
Type    Available   In-Use      Maximum    Largest
chip    233592      282272       515864     76792
fast    341384      182896       524280    197360
total   574976      465168      1040144    197360
```

Ein vollständiger Überblick über das System-RAM wird angezeigt.

Beispiel 2:

```
1> AVAIL CHIP
233592
```

Die Anzahl der freien Bytes im CHIP-RAM wird angezeigt.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls AVAIL.

BREAK

Mit diesem Befehl werden Unterbrechungskennungen im angegebenen Prozeß aktiviert.

Format	BREAK <Prozeß> [ALL | C | D | E | F]
Schablone	PROCESS/A/N,ALL/S,C/S,D/S,E/S,F/S
Pfad	C:

Über den Befehl BREAK werden die angegebenen Unterbrechungskennungen für die angegebene Prozeß-Nummer aktiviert. Mit dem Befehl STATUS können Sie die aktuellen Prozeßnummern anzeigen. Mit C wird die Kennung Ctrl-C aktiviert, mit D die Kennung Ctrl-D usw. Mit der Option ALL werden alle Kennungen von Ctrl-C bis Ctrl-F aktiviert. Standardmäßig wird nur die Kennung Ctrl-C aktiviert.

BREAK hat die gleiche Funktion wie die Auswahl des betreffenden Prozesses durch Anklicken seines Fensters und anschließendem Drücken der Ctrl-Taste und der entsprechenden Buchstabentaste.

Mit Ctrl-C (Vorgabe) wird ein Unterbrechungssignal (BREAK) an einen Prozeß gesendet. Bei einem so abgebrochenen Prozeß wird im Shell-Fenster die Meldung `***Break` angezeigt. Mit Ctrl-D wird die Ausführung einer Befehlsdatei (Skript) abgebrochen. Der Befehl STATUS dient zum Abrufen der aktuellen Prozeßnummern. Die Tastenkombination Ctrl-E ist nicht definiert.

Ctrl-F wird für Programme verwendet, bei denen ein Fenster geöffnet werden kann. Wird an diese Programme Ctrl-F gesendet, wird das zugehörige Fenster geöffnet und vor allen anderen Fenstern plaziert. Nicht alle Programme reagieren auf Ctrl-F.

Beispiel 1:

```
1> BREAK 7
```

Die Unterbrechungskennung Ctrl-C von Prozeß 7 wird aktiviert. Dieselbe Funktion wird erfüllt, wenn Sie Prozeß 7 auswählen und Ctrl-C drücken.

Beispiel 2:

```
1> BREAK 5 D
```

Die Unterbrechungskennung Ctrl-D von Prozeß 5 wird aktiviert.

Siehe auch: STATUS

CD

Mit diesem Befehl wird das aktuelle Verzeichnis gewechselt bzw. angezeigt.

Format CD [<Verzeichnis | Namensmuster>]

Schablone DIR

Pfad Intern

Wenn Sie CD ohne Argumente eingeben, wird der Name des aktuellen Verzeichnisses angezeigt. Wenn Sie zusätzlich den Namen eines gültigen Verzeichnisses angeben, wird dieses zum aktuellen Verzeichnis.

Sie müssen den vollständigen Pfad zum Verzeichnis eingeben, da beim Befehl CD auf der Disk nicht durch alle Unterverzeichnisschachtelungen nach dem Verzeichnis gesucht wird. Wenn sich das angegebene Verzeichnis nicht im aktuellen Verzeichnis bzw. im angegebenen Pfad befindet, erscheint eine entsprechende Fehlermeldung.

Wenn in der Dateistruktur um eine Ebene nach oben gewechselt werden soll (d. h. zum übergeordneten Verzeichnis des aktuellen Verzeichnisses), geben Sie CD gefolgt von einem Leerzeichen und einem Schrägstrich (/) ein. Wenn Sie hinter dem Schrägstrich den Namen eines Verzeichnisses im übergeordneten Verzeichnis angeben, wird zu diesem Verzeichnis gewechselt. Wenn Sie sich im Hauptverzeichnis befinden, bleibt der Befehl **CD /** ohne Wirkung. Sie können auch mehrere, nicht durch Leerzeichen getrennte Schrägstriche eingeben, wobei ein Schrägstrich jeweils für eine nächsthöhere Verzeichnisebene steht.

Wenn Sie direkt in das Hauptverzeichnis des aktuellen Geräts gelangen wollen, geben Sie CD gefolgt von einem Leerzeichen und einem Doppelpunkt ein (`CD :`).

AmigaDOS unterstützt auch implizite CD-Befehle, d. h. Sie können das CD-Befehlswort selbst häufig auslassen. Geben Sie hinter der Eingabeaufforderung lediglich den Verzeichnisnamen, den Pfad, einen Doppelpunkt oder Schrägstriche ein.

Der Befehl CD unterstützt auch Namensmuster. Wenn ein Verzeichnis gefunden wird, das dem angegebenen Namensmuster entspricht, wird es zum aktuellen Verzeichnis. Wenn dem angegebenen Muster mehr als ein Verzeichnis entspricht, wird eine Fehlermeldung angezeigt. Bei impliziten CD-Befehlen können keine Namensmuster verwendet werden. Detaillierte Informationen zu Namensmustern können Sie in Kapitel 3 nachlesen.

Beispiel 1:

```
1> CD DF1:Arbeit
```

Das Verzeichnis "Arbeit" auf der Diskette in Laufwerk DF1: wird zum aktuellen Verzeichnis.

Beispiel 2:

```
1> CD SYS:Com/Basic
```

Das Unterverzeichnis "Basic" im Verzeichnis "Com" von "SYS:" wird zum aktuellen Verzeichnis.

Beispiel 3:

```
1> //
```

Mit diesem impliziten CD-Befehl wird in der Verzeichnisstruktur um zwei Ebenen nach oben gewechselt.

Beispiel 4:

```
1> CD SYS:Li#?
```

Mit dem Muster Li#? wird das Verzeichnis "Libs:" gefunden.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls CD.

CHANGETASKPRI

Mit diesem Befehl wird die Priorität des zur Zeit ausgeführten Prozesses geändert.

Format	CHANGETASKPRI <Priorität> [*PROCESS* <Prozeßnummer>]
Schablone	PRI=PRIORITY/A/N,PROCESS/K/N
Pfad	C:

Mit CHANGETASKPRI wird die Priorität des angegebenen Shell-Prozesses geändert. Wenn keine Prozeßnummer angegeben wurde, bezieht sich der Befehl auf den aktuellen Prozeß. Alle-Shell-Prozesse, die danach aus der <Prozeßnummer> gestartet werden, übernehmen dessen Priorität.

Mit dem Befehl STATUS können Sie die aktuellen Prozeßnummern abrufen.

Für die <Priorität> sind ganze Zahlen im Bereich von -128 bis +127 zulässig, wobei eine größere Zahl für eine höhere Priorität steht (bei einer höheren Priorität wird dem Programm mehr CPU-Zeit zugeordnet). Der Standardwert ist 0. Werte über +10 dürfen aber nicht vergeben werden, da andernfalls wichtige System-Tasks beeinträchtigt werden können.

Beispiel:

```
1> CHANGETASKPRI 4 Process 2
```

Die Priorität von Prozeß 2 wird auf 4 geändert. Alle Programme, die von dieser Shell aus gestartet werden, haben ebenfalls Priorität 4. Sie haben daher Priorität vor allen Benutzer-Tasks, die ohne den Befehl CHANGETASKPRI gestartet wurden (diese Tasks haben Priorität 0).

Siehe auch: STATUS. Kapitel 8 enthält ein weiteres Beispiel zur Verwendung des Befehls CHANGETASKPRI.

COPY

Mit diesem Befehl werden Dateien oder Verzeichnisse kopiert.

Format COPY [FROM] {<Name | Namensmuster>} [TO] <Name> [ALL] [QUIET] [*BUF* | *BUFFER*=<n>] [CLONE] [DATES] [NOPRO] [COM] [NOREQ]

Schablone FROM/M,TO/A,ALL/S,QUIET/S, BUF=BUFFER/K/N,CLONE/S,DATES/S, NOPRO/S,COM/S,NOREQ/S

Pfad C:

Mit COPY wird die im Argument FROM angegebene Datei bzw. das Verzeichnis in die Datei oder das Verzeichnis kopiert, das im Argument TO angegeben wurde. Sie können mehrere Objekte in einem Arbeitsgang kopieren, indem Sie <Name/Namensmuster> entsprechend oft im Argument FROM angeben. Die einzelnen Objekte müssen dabei durch Leerzeichen voneinander getrennt werden. Wenn Sie als FROM-Argument ein Namensmuster oder mehrere Namen angeben, muß es sich beim TO-Argument um ein Verzeichnis handeln.

Wenn der im Argument TO angegebene Dateiname bereits vorhanden ist, wird diese Datei mit der im Argument FROM angegebenen Datei überschrieben. Mit zwei aufeinanderfolgenden doppelten Anführungszeichen ("") können Sie das aktuelle Verzeichnis als Zielverzeichnis angeben. Wenn Sie "" als FROM-Argument verwenden, werden alle Dateien des aktuellen Verzeichnisses kopiert. Geben Sie zwischen die Anführungszeichen kein Leerzeichen ein.

Wenn Sie im Argument FROM ein Verzeichnis angeben, werden nur die Dateien in diesem Verzeichnis kopiert, jedoch nicht die Unterverzeichnisse und die Dateien der Unterverzeichnisse. Sollen die Unterverzeichnisse und die zugehörigen Dateien mitkopiert werden, verwenden Sie die Option ALL. Wenn Sie mehr als eine Datei kopieren, kann beim Kopiervorgang ein Verzeichnis angelegt werden. Geben Sie zum Benennen des neuen Verzeichnisses dessen Namen als letzte Komponente des Pfades im Argument TO ein. Sie können einen beliebigen Namen eingeben. Falls sich das Verzeichnis in einem anderen Pfad befindet, können Sie auch den ursprünglichen Namen wiederverwenden.

Während des Kopiervorgangs werden die Namen der kopierten Dateien auf dem Bildschirm angezeigt. Diese Anzeige wird mit der Option QUIET unterdrückt.

Mit der Option BUF= wird die Anzahl der 512-Byte-Puffer angegeben, die beim Kopiervorgang verwendet werden. (Die Vorgabe ist 128 Puffer, was 64 KB RAM entspricht.) Wenn nach RAM: kopiert wird, sollte die Anzahl der Puffer gering gehalten werden. Bei der Einstellung BUF=0 ist der Puffer so groß wie die zu kopierende Datei.

Standardmäßig werden der im Argument TO angegebenen Datei die Erstellungszeit und das Erstellungsdatum der Kopie zugewiesen, und nicht die Zeit und das Datum verwendet, zu dem die Originaldatei erstellt bzw. zuletzt geändert wurde. Kommentare zum im Argument FROM angegebenen Original werden ignoriert. Schutzbits werden mitkopiert. Sie können diese Vorgaben mit verschiedenen Optionen umdefinieren:

CLONE Uhrzeit und Datum der Erstellung bzw. der letzten Änderung, Kommentare und Schutzbits der FROM-Datei werden mit zur TO-Datei kopiert.

DATES Uhrzeit und Datum der Erstellung bzw. der letzten Änderung der FROM-Datei werden mit zur TO-Datei kopiert.

COM Kommentare der FROM-Datei werden mit zur TO-Datei kopiert.

NOPRO Schutzbits der FROM-Datei werden nicht mit zur TO-Datei kopiert. Die TO-Datei erhält die Standardschutzbits r, w, e und d.

Falls der Kopiervorgang nicht fortgesetzt werden kann, wird in der Regel ein Dialogfenster angezeigt. Wenn Sie jedoch die Option NOREQ angeben, wird die Anzeige von Dialogfenstern unterdrückt. Verwenden Sie diese Option in Befehlsdateien, damit ein Kopierfehler nicht dazu führt, daß die weitere Ausführung der Befehlsdatei gestoppt wird, um auf eine Antwort zu warten. Bei Angabe der Option NOREQ wird in diesem Fall der Befehl COPY abgebrochen und die Verarbeitung der Befehlsdatei fortgesetzt.

Beispiel 1:

```
1> COPY Datei1 TO :Arbeit/Datei2
```

Mit diesem Befehl wird "Datei1" im aktuellen Verzeichnis in das Verzeichnis "Arbeit" im Hauptverzeichnis des aktuellen Geräts kopiert. Dabei wird die Kopie in "Datei2" umbenannt.

Beispiel 2:

```
1> COPY Kapitel#? TO DF1:Sicherung
```

Alle Dateien im aktuellen Verzeichnis, die mit der Zeichenkette "Kapitel" beginnen, werden in das Verzeichnis "Sicherung" auf der der Diskette im Laufwerk "DF1:" kopiert. Falls das Verzeichnis "Sicherung" noch nicht vorhanden ist, wird es jetzt angelegt.

Beispiel 3:

```
1> COPY Arbeit:Test TO ""
```

Mit diesem Befehl werden die Dateien im Unterverzeichnis "Arbeit:Test" in das aktuelle Verzeichnis kopiert. Unterverzeichnisse werden dabei nicht mitkopiert.

Beispiel 4:

```
1> COPY Arbeit:Test TO DF0:Test ALL
```

Mit diesem Befehl werden alle Dateien und Unterverzeichnisse im Unterverzeichnis "Arbeit:Test" in das Verzeichnis "Test" in Laufwerk "DF0:" kopiert. Falls das Verzeichnis "Test" auf dem Laufwerk "DF0:" noch nicht vorhanden ist, wird es jetzt angelegt.

Beispiel 5:

```
1> COPY DF0: TO DF1: ALL QUIET
```

Mit diesem Befehl werden alle Dateien und Verzeichisse von der Diskette in Laufwerk "DF0:" nach "DF1:" kopiert. Dabei werden die Namen der gerade kopierten Dateien und Verzeichnisse nicht angezeigt. (Bei Disketten, die wenig belegt sind, wird dieser Befehl möglicherweise schneller ausgeführt als der Befehl DISKCOPY.) Bei stark fragmentierten Disketten ist so eine Kopie empfehlenswert, um die Zugriffe auf die Dateien spürbar zu beschleunigen.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls COPY.

CPU

Mit diesem Befehl werden Prozessoroptionen eingestellt oder angezeigt.

Format CPU [CACHE | NOCACHE] [BURST | NOBURST] [DATACACHE | NODATACACHE] [DATABURST | NODATABURST] [INSTCACHE | NOINSTCACHE] [INSTBURST | NOINSTBURST] [FASTROM | NOFASTROM] [TRAP | NOTRAP] [COPYBACK | NOCOPYBACK] [EXTERNALCACHE | NOEXTERNALCACHE] [NOMMUTEST] [CHECK 68010 | 68020 | 68030 | 68040 | 68881 | 68882 | 68851 | MMU | FPU]

Schablone CACHE/S,BURST/S,NOCACHE/S,NOBURST/S, DATACACHE/S,NODATACACHE/S, DATABURST/S, NODATABURST/S, INSTCACHE/S,NOINSTCACHE/S, INSTBURST/S,NOINSTBURST/S,COPYBACK/S, NOCOPYBACK/S,EXTERNALCACHE/S, NOEXTERNALCACHE/S,FASTROM/S, NOFASTROM/S,TRAP/S,NOTRAP/S, NOMMUTEST/S,CHECK/K

Pfad C:

Mit CPU können Sie verschiedene Optionen des Amiga-Mikroprozessors an Ihre Erfordernisse anpassen. Außerdem können Sie den Prozessortyp und die zur Zeit aktivierten Optionen abfragen. Ohne ein Argument wird die aktuelle Konfiguration angezeigt.

Viele Optionen stehen nur für bestimmte Modelle der Prozessorserie 680X0 zur Verfügung. Der Prozessor 68020 hat einen speziellen internen Speichertyp, der als Anweisungs-Cache (INSTruction) bezeichnet wird. Bei Verwendung des Anweisungs-Cache werden Anweisungen schneller ausgeführt. Die Prozessoren 68030 und 68040

verfügen über die beiden Cache-Speicher Anweisungs- und Daten-Cache.

Wenn Sie sich gegenseitig ausschließende Optionen angeben, wird die sicherste verwendet. Die Verfügbarkeit der folgenden Optionen hängt vom Typ des jeweiligen Prozessors ab.

CACHE	Alle Cache-Speicher werden aktiviert.
NOCACHE	Alle Cache-Speicher werden inaktiviert.
BURST	Der Burst-Modus wird für Daten und Anweisungen aktiviert.
NOBURST	Der Burst-Modus wird für Daten und Anweisungen inaktiviert.
DATACACHE	Der Daten-Cache wird aktiviert.
NODATACACHE	Der Daten-Cache wird inaktiviert.
DATABURST	Der Burst-Modus wird für Daten aktiviert.
NODATABURST	Der Burst-Modus wird für Daten inaktiviert.
INSTCACHE	Der Anweisungs-Cache wird aktiviert.
NOINSTCACHE	Der Anweisungs-Cache wird inaktiviert.
INSTBURST	Der Burst-Modus wird für Anweisungen aktiviert.
NOINSTBURST	Der Burst-Modus wird für Anweisungen inaktiviert.
FASTROM	Wird von einem Prozessor eine MMU (Memory Management Unit, Speicherzuordnungseinheit) unterstützt, wird das System-ROM in das 32-Bit-RAM kopiert, wodurch der Zugriff auf Betriebssystemfunktionen wesentlich beschleunigt wird. Dieser RAM-Bereich wird dann schreibgeschützt, damit die Daten nicht geändert werden können.
NOFASTROM	FASTROM wird inaktiviert.
TRAP	Nur für Entwickler.
NOTRAP	Nur für Entwickler.
COPYBACK	Der Copyback-Cache des Mikroprozessors 68040 wird aktiviert.
NOCOPYBACK	Der Copyback-Cache des Mikroprozessors 68040 wird inaktiviert.
EXTERNALCACHE	Externer Cache-Speicher wird aktiviert.
NOEXTERNALCACHE	Externer Cache-Speicher wird inaktiviert.

NOMMUTEST Die Einstellungen der MMU können geändert werden, ohne vorher zu prüfen, ob sie gerade verwendet wird.

Wenn Sie die Option CHECK mit einem Schlüsselwort (68010, 68020, 68030, 68040, 68881, 68882 oder 68851, MMU, FPU) angeben, wird geprüft, ob dieser Prozessor vorhanden ist. Wird er (oder ein noch größerer) nicht vorgefunden, wird die in Skripts abfragbare WARN-Bedingung gesetzt.

Beispiele:

```
1> CPU
System: 68030 68881 (INST: Cache Burst) (DATA:
Cache NoBurst)

1> CPU NODATACACHE FASTROM
System: 68030 68881 FastROM (INST: Cache Burst)
(DATA: NoCache NoBurst)

1> CPU NOBURST DATACACHE NOINSTCACHE
System: 68030 68881 (INST: NoCache NoBurst) (DATA:
Cache NoBurst)
```

DATE

Mit diesem Befehl wird das Systemdatum und/oder die Systemzeit eingestellt oder angezeigt.

Format	DATE [<Tag>] [<Datum>] [<Uhrzeit>] [*TO* | *VER* <Dateiname>]
Schablone	DAY,DATE,TIME,TO=VER/K
Pfad	C:

Wenn Sie DATE ohne Argumente eingeben, wird die aktuelle Zeit- und Datumseinstellung des Systems einschließlich des Wochentags angezeigt. Die Zeiteingabe erfolgt im 24-Stunden-Format.

Mit DATE <Datum> wird nur das Datum eingestellt. Das <Datum> wird im Format TT-MMM-JJ (Tag-Monat-Jahr) eingegeben, wobei auch die Bindestriche einzugeben sind. Bei der Datumsangabe ist keine vorangestellte Null erforderlich. Für die Monatsangabe sind die ersten drei Buchstaben des deutschen Monatsnamens einzugeben (die

maximal zweistellige Monatsnummer wird aber auch akzeptiert) und für das Jahr die letzten beiden Ziffern.

Wenn das Datum bereits eingestellt ist, können Sie es durch die Angabe eines Wochentags ändern. Zulässige Argumente für <Tag> sind auch "morgen" und "gestern". Durch die Angabe des (deutschen) Wochentags kann das Datum aber höchstens um sieben Tage vorgestellt werden.

Die <Uhrzeit> wird im Format HH:MM:SS (Stunden:Minuten: Sekunden) eingegeben. Die Sekunden müssen nicht angegeben werden.

Wenn Ihr Amiga über keine batteriegepufferte Echtzeituhr verfügt und Sie kein Datum eingeben, übernimmt das System beim Start das Datum der zuletzt auf der Startdiskette erstellten bzw. geänderten Datei.

Wenn Sie die Option TO oder VER gefolgt von einem Dateinamen angeben, wird die Ausgabe des Befehls DATE in diese Datei gesendet. Wenn bereits eine Datei mit diesem Namen vorhanden ist, wird deren Inhalt überschrieben.

Die mit DATE vorgenommenen Einstellungen wirken sich nur auf die Softwareuhr aus. Sie werden wieder gelöscht, wenn das System ausgeschaltet wird. Soll die batteriegepufferte Echtzeituhr von der Shell aus gestellt werden, nehmen Sie zunächst Einstellungen mit DATE vor. Anschließend verwenden Sie den Befehl SETCLOCK SAVE.

Vom Befehl DATE werden Datum und Uhrzeit zwar nur in einem einzigen Format akzeptiert und angezeigt, aber Programme wie "Clock" (Uhr) können Datum und Uhrzeit im Format des im Landes-Voreinsteller (Locale) ausgewählten Landes anzeigen.

Beispiel 1:

```
1> DATE
6-Sep-92
```

Beispiel 2:

```
1> DATE 6-sep-94
```

Mit diesem Befehl wird das Datum auf den 6. September 1994 gestellt. Die Uhrzeit bleibt unverändert.

Beispiel 3:

```
1> DATE morgen
```

Mit diesem Befehl wird das Datum um einen Tag vorgestellt.

Beispiel 4:

```
1> DATE TO Klaus
```

Mit diesem Befehl wird das aktuelle Datum in die Datei "Klaus" geschrieben.

Beispiel 5:

```
1> DATE 23:00
```

Mit diesem Befehl wird die aktuelle Uhrzeit auf 23.00 eingestellt.

Beispiel 6:

```
1> DATE 1-jan-02
```

Mit diesem Befehl wird das Datum auf den 1. Januar 2002 eingestellt. Das frühestmögliche Datum ist der 1. Januar 1978.

DELETE

Mit diesem Befehl werden Dateien oder Verzeichnisse gelöscht.

Format	DELETE {<Name | Namensmuster>} [ALL] [QUIET] [FORCE]
Schablone	FILE/M/A,ALL/S,QUIET/S,FORCE/S
Pfad	C:

Nach Eingabe von DELETE wird versucht, die angegebenen Elemente zu löschen. Es können mehrere Dateien gleichzeitig gelöscht werden, indem Sie die zu löschenden Dateien einzeln angeben oder

mit Hilfe von Jokerzeichen Namensmuster für eine bestimmte Gruppe zu löschender Dateien verwenden. Die Namensmuster können sich sowohl auf Verzeichnisebenen als auch auf Dateinamen beziehen. Wenn mehrere Dateien gelöscht werden, können Sie den Vorgang mit Ctrl-C abbrechen. Das Löschen mehrerer Dateien wird abgebrochen, wenn ein Objekt gefunden wird, das nicht gelöscht werden kann (z. B. eine löschgeschützte Datei oder eine zur Zeit verwendete Datei). Bei einem DELETE-Befehl mit Namensmuster werden alle löschbaren Dateien gelöscht. Gegebenenfalls werden die Objekte aufgelistet, die nicht gelöscht werden konnten.

Hinweis Vor dem Löschen erfolgt keine Bestätigungsrückfrage. Sie sollten deshalb im Umgang mit Namensmustern sehr gut vertraut sein, bevor Sie sie für den Befehl DELETE verwenden. Außerdem wird empfohlen, von allen wichtigen Dateien stets aktuelle Sicherungskopien anzufertigen. Ansonsten können Sie gelöschte Objekte nicht wiederherstellen.

Wenn Sie versuchen, ein Verzeichnis zu löschen, das Dateien enthält, wird eine diesbezügliche Fehlermeldung angezeigt. Mit der Option ALL können Sie dagegen das angegebene Verzeichnis, die Unterverzeichnisse und alle darin enthaltenen Dateien löschen.

Die Namen der jeweils gelöschten Dateien werden angezeigt. Wenn diese Bildschirmausgabe unterdrückt werden soll, verwenden Sie die Option QUIET.

Wenn das Schutzbit "d" (deletable = löschbar) gelöscht wurde, kann die Datei nur noch durch Angabe der Option FORCE gelöscht werden.

Beispiel 1:

```
1> DELETE Alt-Datei
```

Mit diesem Befehl wird die "Alt-Datei" im aktuellen Verzeichnis gelöscht.

Beispiel 2:

```
1> DELETE Arbeit/Prog1 Arbeit/Prog2 Arbeit
```

Mit diesem Befehl werden die Dateien "Prog1" und "Prog2" im Verzeichnis "Arbeit" und dann das Verzeichnis selbst gelöscht (wenn es keine weiteren Dateien enthält).

Beispiel 3:

```
1> DELETE T#?/#?(1|2)
```

Mit diesem Befehl werden alle Dateien gelöscht, deren Namen auf "1" oder "2" enden und die sich in einem Verzeichnis befinden, dessen Name mit "T" beginnt.

Beispiel 4:

```
1> DELETE DF1:#? ALL FORCE
```

Mit diesem Befehl werden alle Dateien in Laufwerk DF1: gelöscht, einschließlich derjenigen, die als nicht löschbar gekennzeichnet sind.

Siehe auch: PROTECT. Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls DELETE.

DIR

Mit diesem Befehl wird eine sortierte Liste der Dateien in einem Verzeichnis angezeigt.

Format	DIR [<Verzeichnis | Namensmuster>] [*OPT* A | I | AI | D | F] [ALL] [DIRS] [FILES] [INTER]
Schablone	DIR,OPT/K,ALL/S,DIRS/S, FILES/S,INTER/S
Pfad	C:

Die im angegebenen oder aktuellen Verzeichnis enthaltenen Dateien und Verzeichnisse werden angezeigt. Zunächst werden die Verzeichnisse und dann die Dateien in alphabetischer Reihenfolge zweispaltig angezeigt. Mit Ctrl-C können Sie die Auflistung abbrechen.

Folgende Optionen sind möglich:

ALL	Alle Unterverzeichnisse und die dazugehörigen Dateien werden angezeigt.
DIRS	Nur die Verzeichnisse werden angezeigt.
FILES	Nur die Dateien werden angezeigt.
INTER	Der interaktive Modus wird aktiviert.

Die Schlüsselwörter ALL, DIRS, FILES und INTER ersetzen die Optionen OPT A, D, F bzw. I. Diese alten Schlüsselwörter wurden jedoch aus Gründen der Kompatibilität zu früheren AmigaDOS-Versionen noch beibehalten. Verwenden Sie OPT nicht mit den ausgeschriebenen Schlüsselwörtern ALL, DIRS, FILES und INTER.

Im interaktiven Modus wird die Auflistung nach jedem Namen unterbrochen. Ein Fragezeichen wird angezeigt, hinter dem Sie Befehle eingeben können. Folgende Eingaben sind möglich:

Eingabetaste	Der nächste Name wird angezeigt.
E	Es wird in ein Verzeichnis gewechselt, und die darin enthaltenen Dateien werden angezeigt.
B	Es wird in die nächsthöhere Verzeichnisebene zurückgewechselt.
DEL oder DELETE	Eine Datei bzw. ein leeres Verzeichnis wird gelöscht. DEL bezieht sich dabei nicht auf die Del-Taste auf der Tastatur. Es sind hier die Buchstaben D, E und L einzutippen.
T	Der Inhalt der Datei wird angezeigt (wie Type-Befehl).
C oder COMMAND	Zusätzliche AmigaDOS-Befehle können eingegeben werden.
Q	Der interaktive Modus wird inaktiviert.
?	Eine Liste der im interaktiven Modus verfügbaren Befehle wird angezeigt.

Mit der Option COMMAND können Sie fast alle AmigaDOS-Befehle aufrufen, während ein Verzeichnis im interaktiven Modus angezeigt wird. Wenn ein Befehl eingegeben werden soll, geben Sie C (oder COMMAND) hinter der Eingabeaufforderung ? ein. Sie werden dann aufgefordert, den Befehl einzugeben. Geben Sie den gewünschten Befehl ein und drücken Sie die Eingabetaste. Der Befehl wird ausgeführt, und die Auflistung des Verzeichnisses wird fortgesetzt.

Sie können den Befehl auch unmittelbar hinter dem C eingeben, wenn Sie den Befehl in Anführungszeichen setzen.

Beispiel:

```
? C "type prefs.info hex"
```

Anstatt dieses Befehls können Sie auch Q drücken, um den interaktiven Modus zu verlassen, und hinter der normalen Shell-Eingabeaufforderung folgenden Befehl eingeben:

```
1> TYPE Prefs.info HEX
```

Dadurch wird der Inhalt der Datei "Prefs.info" im Hexadezimalformat angezeigt.

Der Befehl FORMAT sollte im interaktiven Modus nicht verwendet werden, da in diesem Fall die Formatierung sofort gestartet wird, ohne daß vorher noch einmal nachgefragt wird. Außerdem darf im interaktiven Modus der Befehl DIR nicht erneut im interaktiven Modus aufgerufen werden, weil dies zu einer fehlerhaften Ausgabe führt.

Beispiel 1:

```
1> DIR Workbench:
```

Mit diesem Befehl werden die Verzeichnisse und Dateien im Hauptverzeichnis der Workbench-Disk angezeigt.

Beispiel 2:

```
1> DIR MeineDisk:#?.memo
```

Mit diesem Befehl werden die Verzeichnisse und Dateien auf "MeineDisk" angezeigt, die auf .memo enden.

Beispiel 3:

```
1> DIR Extras: ALL
```

Mit diesem Befehl wird der gesamte Inhalt der Disk "Extras:" angezeigt, also alle Verzeichnisse, Unterverzeichnisse und alle darin enthaltenen Dateien.

Beispiel 4:

```
1> DIR Workbench: DIRS
```

Mit diesem Befehl werden nur die Verzeichnisse auf "Workbench:" angezeigt.

Beispiel 5:

```
1> DIR Workbench: INTER
```

Mit diesem Befehl wird der Inhalt der Workbench-Disk im interaktiven Modus angezeigt.

Kapitel 8 enthält weitere Beispiele zur Verwendung von DIR.

DISKCHANGE

Mit diesem Befehl wird der Amiga informiert, daß in ein Laufwerk eine andere Disk eingelegt wurde.

Format	DISKCHANGE <Gerät>
Schablone	DRIVE/A
Pfad	C:

Der Befehl DISKCHANGE ist nur dann erforderlich, wenn das Amiga-System 5,25-Zoll-Laufwerke oder Laufwerke für auswechselbare Speichermedien ohne automatische Disk-Wechsel-Erkennung hat. Immer wenn die Diskette oder Kassette eines solchen Laufwerks gewechselt wird, müssen Sie das System mit dem Befehl DISKCHANGE davon in Kenntnis setzen.

Beispiel:

Wenn ein Dialogfenster erscheint und Sie dazu aufgefordert werden, eine neue Diskette in das 5,25-Zoll-Laufwerk DF2: einzulegen, müssen Sie die Diskette einlegen und dann folgenden Befehl eingeben:

```
1> DISKCHANGE DF2:
```

AmigaDOS erkennt dann die neue Diskette, und Sie können die Arbeit fortsetzen.

ECHO

Mit diesem Befehl wird eine Zeichenkette angezeigt.

Format ECHO [<Zeichenkette>] [NOLINE] [*FIRST* <n>] [*LEN* <n>] [*TO* <Dateiname>]

Schablone /M,NOLINE/S,FIRS/K/N,LEN/K/N,TO/K

Pfad Intern

Vom Befehl ECHO wird die angegebene Zeichenkette an das aktuelle Ausgabefenster bzw. -gerät übertragen. Standardmäßig ist dies der Bildschirm. Über die Option TO kann die Zeichenkette jedoch an das angegebene Gerät bzw. die angegebene Datei umgeleitet werden.

Bei Angabe der Option NOLINE wird der Cursor nicht automatisch in die nächste Zeile gesetzt, nachdem die Zeile ausgegeben wurde.

Mit den Optionen FIRST und LEN können Sie eine Unterzeichenkette ausgeben. FIRST <n> gibt die Position des ersten auszugebenden Zeichens an. LEN <n> gibt an, wie viele Zeichen (ab der ersten Position) ausgegeben werden. Wenn die Option FIRST weggelassen und nur das Schlüsselwort LEN verwendet wird, besteht die Zeichenkette aus den rechten <n> Zeichen der Hauptzeichenkette. Wenn die Zeichenkette z. B. 20 Zeichen lang ist und Sie für LEN 4 angeben, werden das 17., 18., 19. und 20. Zeichen aus der Kette übertragen.

Beispiele:

```
1> ECHO "Hallo Leute!"
Hallo Leute!

1> ECHO "Hallo Leute!" NOLINE FIRST 0 LEN 5
Hallo1>
```

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls ECHO.

ED

Mit diesem Befehl wird der Texteditor ED (Bildschirmeditor) zum Edieren von Textdateien aufgerufen.

Format ED [FROM] <Dateiname> [SIZE <n>] [*WITH* <Dateiname>] [*WINDOW* <Fensterspezifikation>] [TABS <n>] [WIDTH | COLS <n>] [HEIGHT | ROWS <n>]

Schablone FROM/A,SIZE/N,WITH/K,WINDOW/K, TABS/N,WIDTH=COLS/N,HEIGHT=ROWS/N

Pfad C:

Weitere Informationen zum Texteditor ED können Sie Kapitel 4 entnehmen. Kapitel 8 enthält ein Beispiel zur Verwendung des Befehls ED.

EDIT

Mit diesem Befehl wird der Zeileneditor EDIT aufgerufen, der zum Edieren von Textdateien durch sequentielle Bearbeitung der Quellendateien dient.

Format EDIT [FROM] <Dateiname> [[TO] <Dateiname>] [*WITH* <Dateiname>] [*VER* <Dateiname>][*OPT* P <Zeilen> | W <Zeichen> | P<Zeilen>W<Zeichen>] [WIDTH <Zeichen>] [PREVIOUS <Zeilen>]

Schablone FROM/A,TO,WITH/K,VER/K,OPT/K, WIDTH/N,PREVIOUS/N

Pfad C:

Weitere Informationen zum Zeileneditor EDIT können Sie Kapitel 4 entnehmen.

ELSE

Mit diesem Befehl wird eine Alternative für eine IF-Anweisung in einer Befehlsdatei angegeben.

Format ELSE

Schablone (keine)

Pfad Intern

Der Befehl ELSE muß in Verbindung mit einem IF-Befehl verwendet werden. ELSE wird in einem IF-Block in einer Befehlsdatei benutzt, damit bei Nichterfüllung der Bedingung eine entsprechende andere Anweisung ausgeführt wird. Wenn die Bedingung falsch ist, wird die Ausführung der Befehlsdatei in der Zeile unter ELSE fortgesetzt. Die dazwischenliegenden Befehle werden übersprungen. Wenn die Bedingung wahr ist, werden die auf die Anweisung IF folgenden Befehle bis zur Anweisung ELSE ausgeführt. Anschließend wird die Ausführung bei der Anweisung ENDIF fortgesetzt, mit der der IF-Block abgeschlossen ist.

Beispiel:

Die Befehlsdatei "Anzeige" enthält folgenden Block:

```
IF exists Bilddatei
   MultiView Bilddatei
ELSE
   ECHO "Bilddatei nicht in diesem Verzeichnis."
ENDIF
```

Wenn sich die Datei "Bilddatei" im aktuellen Verzeichnis befindet, wird das Programm MultiView aufgerufen und die "Bilddatei" angezeigt.

Wenn sich die Datei "Bilddatei" nicht im aktuellen Verzeichnis befindet, wird die Ausführung der Befehlsdatei bei der Anweisung ECHO fortgesetzt. Folgende Meldung wird im Shell-Fenster angezeigt:

```
Bilddatei nicht in diesem Verzeichnis.
```

Siehe auch: IF, ENDIF, EXECUTE

ENDCLI

Mit diesem Befehl wird ein Shell-Prozeß beendet.

Format	ENDCLI
Schablone	(keine)
Pfad	Intern

Mit ENDCLI wird ein Shell-Prozess beendet.

Siehe auch: ENDSHELL

ENDIF

Mit diesem Befehl wird ein IF-Block in einer Befehlsdatei beendet.

Format ENDIF

Schablone (keine)

Pfad Intern

Der Befehl ENDIF ist bei Verwendung von IF-Befehlen erforderlich. ENDIF wird in Befehlsdateien zum Abschluß eines IF-Blocks verwendet. Wenn die Bedingung falsch ist bzw. wenn die Bedingung wahr ist und nach der Ausführung der zugehörigen Befehle die Anweisung ELSE angetroffen wird, wird die Ausführung der Befehlsdatei beim nächsten Befehl ENDIF fortgesetzt. Die Anweisung IF muß immer mit ENDIF abgeschlossen sein.

ENDIF bezieht sich jeweils auf den letzten IF- oder ELSE-Befehl.

Siehe auch: IF, ELSE. Kapitel 8 enthält Beispiele zur Verwendung des Befehls ENDIF.

ENDSHELL

Mit diesem Befehl wird ein Shell-Prozeß beendet.

Format ENDSHELL

Schablone (keine)

Pfad Intern

ENDSHELL dient zum Beenden eines Shell-Prozesses und zum Schließen des zugehörigen Shell-Fensters.

Shell-Prozesse können auch mit dem Befehl ENDCLI, durch Anklicken des Schließsymbols oder durch Drücken der Tastenkombination Ctrl-\ beendet werden.

ENDSHELL sollte nur verwendet werden, wenn die Workbench geladen oder eine weitere Shell aktiv ist. Wenn Sie die Workbench

verlassen haben und dann die einzige Shell schließen, können Sie erst nach einem Neustart wieder mit dem Amiga kommunizieren.

Das Shell-Fenster kann nicht geschlossen werden, wenn Prozesse, die von dieser Shell aus aufgerufen wurden, noch nicht abgearbeitet sind. Obwohl das Fenster geöffnet bleibt, werden keine neuen Eingaben angenommen. Die Prozesse müssen beendet werden, bevor das Fenster geschlossen wird. Wenn Sie z. B. einen Editor von der Shell aus öffnen, wird das Shell-Fenster erst dann geschlossen, wenn Sie den Editor verlassen.

Kapitel 8 enthält Beispiele zur Verwendung des Befehls ENDSHELL.

ENDSKIP

Mit diesem Befehl wird ein SKIP-Block in einer Befehlsdatei beendet.

Format ENDSKIP

Schablone (keine)

Pfad Intern

ENDSKIP wird in Befehlsdateien zum Beenden der Ausführung eines SKIP-Blocks verwendet. Mit einem SKIP-Block können Sie Befehle überspringen, wenn eine bestimmte Bedingung erfüllt ist. Wenn der Befehl ENDSKIP angetroffen wird, wird die Ausführung der Befehlsdatei in der Zeile unter diesem Befehl fortgesetzt. Das Bedingungskennzeichen wird auf 5 (WARN) gesetzt.

Siehe auch: SKIP

EVAL

Mit diesem Befehl werden ganzzahlige oder Boolesche Ausdrücke ausgewertet.

Format EVAL <Wert1> {[<Operation>] [<Wert2>]} [*TO* <Datei>] [*LFORMAT*=<Zeichenkette>]

Schablone VALUE1/A,OP,VALUE2/M,TO/K,LFORMAT/K

Pfad C:

Mit EVAL werden ganzzahlige Ausdrücke ausgewertet, und das Ergebnis wird ausgegeben. Stellen hinter dem Komma werden ggf. bei Eingabewerten und Ergebnissen abgeschnitten. Wird keine Ganzzahl als Eingabewert verwendet, endet die Auswertung mit dem Dezimalzeichen.

<Wert1> und <Wert2> können Dezimal-, Hexadezimal- oder Oktalzahlen sein, wobei Dezimalzahlen die Vorgabe sind. Hexadezimalzahlen werden mit vorangestelltem 0X oder #X gekennzeichnet. Oktalzahlen werden 0 oder # vorangestellt. Buchstaben werden durch einen vorangestellten Apostroph (') gekennzeichnet und ihrem ASCII-Wert entsprechend ausgewertet.

Mit dem Schlüsselwort LFORMAT können Sie das Ausgabeformat für das Ergebnis definieren. Sie können %X (hexadezimal), %O (oktal), %N (dezimal) oder %C (Zeichen) angeben. Bei den Optionen %X und %O muß außerdem die Anzahl der Stellen angegeben werden (bei Angabe von %X8 wird z. B. eine 8stellige Hexadezimalzahl ausgegeben). Soll eine neue Zeile erzeugt werden, geben Sie in der Zeichenkette hinter LFORMAT *N ein.

Die unterstützten Operationen und die zugehörigen Symbole finden Sie in der folgenden Tabelle.

Operation	Symbol
Addition	+
Subtraktion	-
Multiplikation	*
Division	/
Modulo	mod, M, m oder %
AND (bitweise)	&
OR (bitweise)	|
NOT (bitweise)	~
Shift links	lsh, L oder l
Shift rechts	rsh, R oder r
Negation	-
Exklusives OR	xor, X oder x
Bitweise Übereinstimmung	eqv, E oder e

Vorgabewert 10 festlegen. Wenn Sie den Wert erhöhen, geben Sie damit an, daß eine bestimmte Kategorie von Fehlern nicht als schwerwiegend angesehen werden soll und daß die Ausführung nachfolgender Befehle nach dem Auftreten solcher Fehler fortgesetzt werden soll. Das Argument muß eine positive Zahl sein. Beim Verlassen der Befehlsfolge wird der Fehlergrenzwert wieder auf den anfänglichen Wert von 10 zurückgesetzt.

Wenn Sie kein Argument angeben, wird der aktuelle Fehlergrenzwert angezeigt.

Beispiel:

Eine Befehlsdatei enthält folgende Zeilen:

```
COPY DF0:MeineDatei to RAM:
ECHO "MeineDatei wird kopiert."
```

Wird "MeineDatei" nicht gefunden, wird die Befehlsdatei abgebrochen, und im Shell-Fenster erscheint folgende Meldung:

```
COPY: Objekt nicht gefunden
COPY failed returncode 20:
```

Wenn Sie aber den Fehlergrenzwert auf einen höheren Wert als 20 eingestellt haben, wird die Befehlsdatei weiter ausgeführt, selbst wenn der Befehl COPY nicht ausgeführt werden kann:

```
FAILAT 21
COPY DF0:MeineDatei to RAM:
ECHO "MeineDatei wird kopiert."
```

Folgende Meldung wird angezeigt:

```
COPY: Objekt nicht gefunden
MeineDatei wird kopiert.
```

Siehe auch: ECHO, EXECUTE.

FAULT

Mit diesem Befehl werden die Meldungstexte für den angegebenen Fehlercode ausgegeben.

Format FAULT {<n>}

Schablone /N/M

Pfad Intern

Mit FAULT werden die den angegebenen Fehlercodes entsprechenden Meldungstexte ausgegeben. Dabei können Sie beliebig viele durch Leerzeichen voneinander getrennte Fehlercodes angeben, deren Meldungstexte gleichzeitig ausgegeben werden sollen.

Beispiel:

Folgende Fehlermeldung wurde angezeigt:

```
Error when opening DF1:TestDatei 205
```

Wenn Sie nähere Informationen benötigen, geben Sie folgenden Befehl ein:

```
1> FAULT 205
FAULT 205: Objekt nicht gefunden
```

Damit wird der Grund für den Fehler angegeben: "TestDatei" konnte nicht gefunden werden.

Anhang A enthält eine vollständige Liste der Fehlermeldungen.

FILENOTE

Mit diesem Befehl wird einer Datei ein Kommentar hinzugefügt.

Format FILENOTE [FILE] <Datei | Namensmuster> [COMMENT <Kommentar>] [ALL] [QUIET]

Schablone FILE/A,COMMENT,ALL/S,QUIET/S

Pfad C:

Mit FILENOTE können Sie der angegebenen Datei oder allen Dateien, die dem angegebenen Muster entsprechen, einen beliebigen Kommentar mit einer maximalen Länge von 79 Zeichen hinzufügen.

Wenn der <Kommentar> Leerzeichen enthält, muß er in doppelte Anführungszeichen gesetzt werden. Wenn im eigentlichen Kommentar Anführungszeichen verwendet werden sollen, müssen Sie diesen jeweils ein Sternchen (*) voranstellen. Der gesamte Kommentar muß dann zwischen Anführungszeichen stehen, auch wenn er keine Leerzeichen enthält.

Wenn Sie das Argument <Kommentar> nicht angeben, werden alle Kommentare der angegebenen Datei gelöscht.

Das Erstellen eines Kommentars mit FILENOTE entspricht der Eingabe eines Kommentars im Feld "Kommentar" des Fensters "Information" für das Piktogramm. Änderungen, die mit FILENOTE vorgenommen wurden, werden im Informationsfenster angezeigt und umgekehrt.

Mit dem Befehl LIST können Sie über FILENOTE erstellte Kommentare auflisten. Hat eine Datei Kommentare, werden diese bei Verwendung des Befehls LIST unterhalb des Dateinamens angezeigt.

Wenn mit dem Argument TO des Befehls COPY Daten in eine bestehende Datei kopiert werden, wird diese überschrieben; der ursprüngliche Kommentar bleibt jedoch erhalten. Kommentare, die zu der mit FROM angegebenen Datei gehören, werden nur mitkopiert, wenn Sie den Befehl COPY mit der Option CLONE oder COM verwenden.

Bei Angabe der Option ALL wird der <Kommentar> allen Dateien und Unterverzeichnissen angefügt, deren Namen dem angegebenen Namensmuster entsprechen. Wenn Sie die Option QUIET angeben, wird die Bildschirmausgabe unterdrückt.

Beispiel 1:

```
1> FILENOTE Sonata "allegro non troppo"
```

Mit diesem Befehl wird der Datei "Sonata" der Kommentar **allegro non troppo** hinzugefügt.

Beispiel 2:

```
1> FILENOTE Toccata "*"presto*""
```

Mit diesem Befehl wird der Datei "Toccata" der Kommentar **"presto"** hinzugefügt.

GET

Mit diesem Befehl wird der Wert einer lokalen Variablen abgerufen.

Format	GET <Name>
Schablone	NAME/A
Pfad	Intern

Mit GET wird der Wert einer lokalen Umgebungsvariable abgerufen und im aktuellen Fenster angezeigt.

Lokale Umgebungsvariablen können nur von der Shell interpretiert werden, in der sie definiert wurden, sowie von Shells, die mit dem Befehl NEWSHELL in der Ausgangs-Shell gestartet wurden. Wenn Sie eine Shell über das Shell-Piktogramm oder über den Menüpunkt "Befehl ausführen" öffnen, sind vorher definierte lokale Umgebungsvariablen nicht verfügbar.

Beispiel:

```
1> GET editor
Extras:Tools/MEmacs
```

Siehe auch: SET

GETENV

Mit diesem Befehl wird der Wert einer globalen Variable abgerufen.

Format	GETENV <Name>
Schablone	NAME/A
Pfad	Intern

Mit GETENV wird der Wert einer globalen Umgebungsvariable abgefragt und im aktuellen Fenster angezeigt. Globale Variablen werden als Klartextdateien im Verzeichnis "ENV:" gespeichert und können in allen Shells verwendet werden.

Beispiel:

```
1> GETENV editor
Extras:Tools/MEmacs
```

Siehe auch: SETENV

ICONX

Hinweis ICONX wird als Standardprogramm für Projektpiktogramme verwendet. ICONX kann nicht als Shell-Befehl aufgerufen werden.

Bei Verwendung von ICONX kann eine Befehlsdatei mit AmigaDOS-Befehlen durch Anklicken des zugehörigen Piktogramms aufgerufen werden.

Format ICONX

Schablone (keine)

Pfad C:

Wenn ICONX verwendet werden soll, müssen Sie ein Projekt-Piktogramm für die Befehlsdatei erstellen oder kopieren. Öffnen Sie das Fenster "Information" des Piktogramms und ändern Sie das Standardprogramm des Piktogramms auf C:ICONX. Anschließend wählen Sie "Speichern" aus, um die geänderte .info-Datei zu speichern. Die Befehlsdatei kann jetzt aufgerufen werden, indem Sie das Piktogramm doppelt anklicken.

Wenn das Piktogramm geöffnet wird, wechselt ICONX vor der Ausführung der Befehlsdatei in das Verzeichnis, das das Projekt-Piktogramm enthält. Auf der Workbench wird ein Eingabe-/Ausgabefenster für die Befehlsdatei geöffnet, wenn von der Befehlsdatei Ausgabedaten generiert werden.

Im Piktogramm für die Befehlsdatei können mehrere Merkmale angegeben werden. Mit dem Merkmal WINDOW können Sie z. B. andere Angaben zum Eingabe-/Ausgabefenster machen. Standardmäßig werden folgende Angaben verwendet:

```
WINDOW=CON:0/50//80/IconX/AUTO/WAIT/CLOSE
```

Aufgrund der Option AUTO wird ein Fenster nur geöffnet, wenn tatsächlich eine Ausgabe durch die Befehlsdatei erfolgt. Wenn ein Fenster geöffnet wird, bleibt es wegen der Option WAIT auch nach Beendigung der Befehlsdatei so lange geöffnet, bis das Fenster vom Benutzer geschlossen wird. Durch Angabe der Option CLOSE wird ein Schließsymbol im Fenster hinzugefügt.

Mit dem eigenen Merkmal WAIT (nicht mit der Option WAIT des Merkmals WINDOW zu verwechseln) können Sie angeben, wie viele Sekunden das Eingabe-/Ausgabefenster nach Beendigung der Befehlsdatei geöffnet bleiben soll. Wenn Sie diese Option verwenden, können Standardein- bzw. -ausgabefenster erst nach Ablauf des mit WAIT angegebenen Zeitraums geschlossen werden. Das Merkmal DELAY besitzt die gleiche Funktion, sein Parameter wird aber in Einheiten von 1/50 Sekunden anstatt in Sekunden angegeben.

Mit dem Merkmal STACK können Sie angeben, wie viele Bytes bei der Ausführung der Befehlsdatei für den Stack verwendet werden sollen. Wird dieses Merkmal nicht verwendet, wird die Vorgabe 4096 Byte verwendet.

Bei Verwendung des Merkmals USERSHELL wird die Befehlsdatei unter Verwendung der aktuellen Benutzer-Shell und nicht unter Verwendung der normalen ROM-Shell ausgeführt. Geben Sie dazu USERSHELL=YES ein. Bei Benutzer-Shells handelt es sich um Shell-Software von anderen Herstellern, die Sie anstatt der mitgelieferten standardmäßigen Shell-Umgebung auf Ihrem System installieren können.

Mit der erweiterten Auswahl können Sie Dateien, die Piktogramme haben, als Argumente an die Befehlsdatei übergeben. Die Dateinamen werden von der Befehlsdatei als Schlüsselwörter verwendet. Dazu muß am Anfang der Befehlsdatei die Zeile .KEY stehen. In diesem Fall wird die Befehlsdatei mit dem AmigaDOS-Befehl EXECUTE aufgerufen.

Siehe auch: EXECUTE. Kapitel 8 enthält Beispiele zur Verwendung des Befehls ICONX.

IF

Mit diesem Befehl werden bedingte Operationen in Befehlsdateien ausgewertet.

Format	IF [NOT] [WARN | ERROR | FAIL] [<Zeichenkette> *EQ | GT | GE* <Zeichenkette>] [VAL] [*EXISTS* <Dateiname>]
Schablone	NOT/S,WARN/S,ERROR/S,FAIL/S,EQ/K,GT/K, GE/K,VAL/S,EXISTS/K
Pfad	Intern

In einer Befehlsdatei werden mit IF, vorausgesetzt die angegebene Bedingung ist wahr, die nachfolgenden Befehle bis zu den Befehlen ENDIF oder ELSE ausgeführt. IF muß in Verbindung mit ENDIF verwendet werden, ELSE kann verwendet werden. Wenn die Bedingung falsch ist, wird die Ausführung direkt nach dem Befehl ENDIF bzw. ELSE fortgesetzt. Auf IF und ELSE dürfen mehrere Befehlszeilen bis zum abschließenden ENDIF folgen.

Bei einer Verschachtelung von IF-Befehlen wird die Verarbeitung beim enstprechenden ENDIF fortgesetzt.

Die weiteren Schlüsselwörter haben folgende Funktion::

NOT	Die Auswertung des Ergebnisses wird umgekehrt.
WARN	Wahr, wenn vorhergehender Rückgabecode >= 5
ERROR	Wahr, wenn vorhergehender Rückgabecode größer gleich 10. Nur verfügbar, wenn FAILAT auf größer als 10 eingestellt ist.
FAIL	Wahr, wenn vorhergehender Rückgabecode größer gleich 20. Nur verfügbar, wenn FAILAT auf größer als 20 eingestellt ist.
<a> GT <b>	Wahr, wenn Text a größer Text b (ohne Unterscheidung von Groß- und Kleinschreibung). Für ≤ geben Sie NOT GT an.
<a> GE <b>	Wahr, wenn Text a größer gleich Text b (ohne Unterscheidung von Groß- und Kleinschreibung). Für kleiner geben Sie NOT GE an.

<a> EQ <b>	Wahr, wenn Text a = Text b (ohne Unterscheidung von Groß- und Kleinschreibung).
VAL	Gibt numerischen Vergleich an.
EXISTS <file>	Wahr, wenn Datei oder Verzeichnis vorhanden.

Wenn mehr als ein Schlüsselwort drei Bedingungskennzeichen (WARN, ERROR, FAIL) angegeben wird, wird das mit dem niedrigsten Wert verwendet.

Sie können mit dem Befehl IF lokale oder globale Variablen verwenden, wenn Sie dem Variablennamen ein Dollarzeichen ($) voranstellen.

Beispiel 1:

```
IF EXISTS Arbeit/Prog
   TYPE Arbeit/Prog HEX
ELSE
   ECHO "Nicht vorhanden"
ENDIF
```

Wenn sich die Datei "Arbeit/Prog" im aktuellen Verzeichnis befindet, wird ihr Inhalt angezeigt. Andernfalls erscheint die Meldung **"Nicht vorhanden"**, und die Ausführung wird hinter ENDIF fortgesetzt.

Beispiel 2:

```
IF ERROR
   SKIP errlab
ENDIF
ECHO "Kein Fehler"
LAB errlab
```

Wenn der vorhergehende Befehl einen Rückgabecode von 10 oder größer erzeugt hat, wird der Befehl ECHO übersprungen. Die Ausführung wird bei der Sprungmarke "errlab" fortgesetzt.

Siehe auch: ELSE, ENDIF, EXECUTE, FAILAT, LAB, QUIT, SKIP. Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls IF.

INFO

Mit diesem Befehl werden Informationen zu angemeldeten Geräten angezeigt.

Format INFO [<Gerät>]

Schablone DEVICE

Pfad C:

Mit INFO wird eine Informationszeile zu allen angemeldeten Speichergeräten einschließlich Diskettenlaufwerken und Festplattenpartitionen angezeigt. Diese Zeile enthält folgende Angaben: Einheitenname (Unit), maximale Speicherkapazität der Disk (Size), belegte (Used) und freie (Free) Speicherblöcke, Prozentsatz des belegten Speicherplatzes (Full), Anzahl der aufgetretenen behebbaren Disk-Fehler (Errs) sowie Status und Name der Disk.

Bei Angabe des Arguments <Gerät> werden die angezeigten Informationen auf ein Gerät bzw. auf einen Datenträger beschränkt.

Beispiel:

```
1>INFO
Unit Size Used Free  Full Errs Status     Name
DF0: 879K 1738   20  98%  0   Read Only  Workbench
DF1: 879K  418 1140  24%  0   Read/Write Text-6

Volumes available:
Workbench [Mounted]
Text-6 [Mounted]
```

INSTALL

Mit diesem Befehl wird ein Boot-Block auf einer formatierten Diskette oder PCMCIA-Karte geschrieben oder überprüft, wobei angegeben wird, ob es sich um einen startfähigen Block handelt.

Format INSTALL [DRIVE] <DF0: | DF1: | DF2: | DF3: | CC0:> [NOBOOT] [CHECK] [FFS]

Schablone DRIVE/A,NOBOOT/S,CHECK/S,FFS/S

Pfad C:

Mit INSTALL wird der Boot-Block einer Diskette oder einer PCMCIA-Speicherkarte gelöscht, und ein neuer Boot-Block wird darauf geschrieben. INSTALL hat keine Auswirkungen auf Dateien oder Verzeichnisse, die sich auf der Diskette bzw. Speicherkarte befinden. Zum erfolgreichen Systemstart müssen weiterhin die erforderlichen Dateien und Verzeichnisse auf dem Gerät vorhanden sein.

Mit der Option NOBOOT wird der Boot-Block von einer AmigaDOS-Diskette bzw. einer -Speicherkarte gelöscht, so daß sie nicht mehr startfähig ist.

Mit der Option CHECK wird die Diskette bzw. die Speicherkarte auf einen gültigen Boot-Block überprüft. INSTALL meldet, ob die Diskette bzw. die Speicherkarte startfähig ist und ob sich der standardmäßige Commodore-Amiga-Boot-Block auf dem Datenträger befindet. Dies ist bei der Suche nach einigen Viren hilfreich.

Der Schalter FFS (FastFileSystem) wird nicht mehr unterstützt. Er wurde aber aus Gründen der Kompatibilität zu früheren Befehlsdateien und Programmen in der Befehlsschablone beibehalten.

Beispiel 1:

```
1> INSTALL DF0: CHECK
No bootblock installed (Kein Boot-Block
installiert)
```

Dies zeigt an, daß die Diskette in Laufwerk DF0: nicht startfähig ist.

Beispiel 2:

```
1> INSTALL DF0:
```

Mit diesem Befehl wird die Diskette in Laufwerk DF0: startfähig gemacht.

Beispiel 3:

```
1> INSTALL DF0: CHECK
Appears to be FFS bootblock (Wahrscheinlich FFS-
Boot-Block)
```

Dies zeigt an, daß sich in Laufwerk DF0: eine FFS-Diskette befindet.

JOIN

Mit diesem Befehl werden mindestens zwei Dateien zu einer neuen Datei verknüpft.

Format JOIN [FILE] {<Datei|Namensmuster>} *AS|TO* <Dateiname>

Schablone FILE/M/A,AS=TO/K/A

Pfad C:

Mit JOIN werden die aufgelisteten Dateien in der angegebenen Reihenfolge in eine neue, gemeinsame Datei kopiert. Sie müssen einen Zieldateinamen eingeben, der mit mit keinem der Namen der Quelldateien übereinstimmt. Die Ausgangsdateien werden durch diesen Befehl nicht verändert. Mit dem Befehl JOIN können beliebig viele Dateien in einem Arbeitsgang verknüpft werden.

Anstelle von AS kann auch TO verwendet werden.

Beispiel:

```
1> JOIN Teil1 Teil2 Teil3 AS Textdatei
```

Kapitel 8 enthält ein weiteres Beispiel zur Verwendung des Befehls JOIN.

LAB

Mit diesem Befehl wird eine Sprungmarke in einer Befehlsdatei angegeben.

Format LAB [<Zeichenkette>]

Schablone (keine)

Pfad Intern

Mit LAB wird in einer Befehlsdatei eine Sprungmarke (engl.: label) festgelegt, nach der der Befehl SKIP suchen kann. Die <Zeichenkette> für die Sprungmarke kann beliebig lang sein, muß aber aus alphanumerischen Zeichen bestehen. Sonderzeichen sind

nicht zulässig. Wenn die Zeichenkette Leerzeichen enthält, muß sie in Anführungszeichen gesetzt werden.

Siehe auch: SKIP, IF, EXECUTE. Kapitel 8 enthält Beispiele zur Verwendung des Befehls LAB 8.

LIST

Mit diesem Befehl werden bestimmte Informationen über Verzeichnisse und Dateien angezeigt.

Format LIST [{<Verzeichnis | Namensmuster | Dateiname>}] [*P* | *PAT* <Namensmuster>] [KEYS] [DATES] [NODATES] [*TO* <Name>] [*SUB* <Zeichenkette>] [*SINCE* <Datum>] [*UPTO* <Datum>] [QUICK] [BLOCK] [NOHEAD] [FILES] [DIRS] [*LFORMAT* <Zeichenkette>] [ALL]

Schablone DIR/M,P=PAT/K,KEYS/S,DATES/S,NODATES/S, TO/K,SUB/K,SINCE/K,UPTO/K,QUICK/S, BLOCK/S,NOHEAD/S,FILES/S,DIRS/S, LFORMAT/K,ALL/S

Pfad C:

Mit LIST werden Informationen über den Inhalt des aktuellen Verzeichnisses angezeigt. Wenn Sie als Argument <Verzeichnis>, <Namensmuster> oder <Dateiname> angeben, werden mit LIST Informationen über das angegebene Verzeichnis, über alle Verzeichnisse und Dateien, die dem Namensmuster entsprechen, bzw. über die angegebene Datei angezeigt. Mit dem Argument PAT können Sie ein zusätzliches Namensmuster angeben.

Wenn Sie keine anderen Optionen angeben, wird mit LIST folgendes angezeigt:

Name Name der Datei bzw. des Verzeichnisses.

Größe Größe der Datei in Byte. Wenn die Datei keine Daten enthält, steht in diesem Feld "empty" (leer). Bei Verzeichnissen steht hier "Dir" (Verzeichnis).

Schutzbits	Die Kennbuchstaben der für diese Datei aktivierten Schutzbits werden angezeigt. Nicht aktivierte Bits sind als Gedankenstrich dargestellt. Für die meisten Dateien werden die vorgegebenen Schutzbits "----rwed" für readable/writable/executable/deleteable (lesbar/schreibbar /ausführbar/löschbar) angezeigt. Weitere Informationen zu Schutzbits finden Sie unter dem Befehl PROTECT.
Datum und Uhrzeit	Datum und Uhrzeit der Erstellung der Datei oder der letzten Änderung.
Kommentar	Kommentar, der dieser Datei mit dem Befehl FILENOTE hinzugefügt wurde. Vor dem Kommentar steht ein Doppelpunkt (:).

Mit folgenden Optionen des Befehls LIST kann das Erscheinungsbild der Ausgabe geändert werden:

KEYS	Die Blocknummer jedes Dateikopfs und Verzeichnisses auf dem Datenträger wird angezeigt.
DATES	Die Daten werden angezeigt (z. B. im Format TT-MMM-JJ).
NODATES	Datum und Uhrzeit werden nicht angezeigt.
TO <Name>	Eine Ausgabedatei für LIST wird angegeben. Standardmäßig erfolgt die Ausgabe im aktuellen Fenster.
SUB <Zkette>	Nur Dateien, deren Namen diese <Zeichenkette> enthalten, werden angezeigt.
SINCE <Datum>	Nur Dateien, die seit dem angegebenen Datum erstellt wurden, werden angezeigt. Datum und ggf. Wochentage sind in Deutsch anzugeben.
UPTO <Datum>	Nur Dateien, die am oder vor dem angegeben Datum erstellt wurden, werden angezeigt.
QUICK	Nur die Namen von Dateien und Verzeichnissen werden angezeigt.
BLOCK	Die Dateigröße wird in 512-Byte-Blöcken anstatt in Byte angegeben.
NOHEAD	Die Kopfzeile der Liste und die Zusammenfassung werden nicht angezeigt.
FILES	Nur Dateien werden angezeigt (keine Verzeichnisse).
DIRS	Nur Verzeichnisse werden angezeigt (keine Dateien).
LFORMAT <Zkette>	Das spezifische Format der Ausgabe wird durch eine Zeichenkette definiert.

ALL	Alle Dateien im aktuellen Verzeichnis und in den Unterverzeichnissen werden angezeigt.

Mit der Option LFORMAT wird die Ausgabe des Befehls LIST geändert. Außerdem kann diese Option zum automatischen Generieren von Befehlsdateien verwendet werden. Bei der Verwendung von LFORMAT müssen Sie eine Zeichenkette zur Festlegung des Ausgabeformats mitgeben. Diese Zeichenkette wird als Ausgabe für alle in der Regel aufgelisteten Dateien und Verzeichnisse verwendet. Die Zeichenkette kann den von Ihnen angegebenen Text und zusätzlich die üblichen Ausgabedaten des Befehls LIST enthalten. Wenn Sie LFORMAT angeben, werden die Option QUICK und NOHEAD automatisch ausgewählt. Soll die Ausgabe gespeichert werden, müssen Sie sie mit dem Operator > in eine Datei umleiten oder das Argument TO mit einem Dateinamen angeben. (Kapitel 8 enthält Beispiele zur Verwendung der Option LIST LFORMAT.)

In der Zeichenkette verwendbare Platzhalter:

%A	Dateiattribute werden ausgegeben (Schutzbits).
%B	Die Größe der Datei wird in Blöcken ausgegeben.
%C	Der Kommentar der Datei wird ausgegeben.
%D	Das Datum der Datei wird ausgegeben.
%E	Nur die Dateinamenserweiterung wird ausgegeben.
%K	Die Datei-Header-Blocknummer wird ausgegeben.
%L	Die Länge der Datei in Byte wird ausgegeben.
%M	Nur der Dateiname wird eingefügt (ohne Erweiterung).
%N	Der volle Name der Datei wird ausgegeben.
%P	Der Pfad zur Datei wird ausgegeben.
%S	Dieser Operator wurde durch %N und %P ersetzt, ist aber weiterhin funktionsfähig.
%T	Die Erstellungszeit der Datei wird ausgegeben.

Sie können zwischen dem Prozentzeichen (%) und der Feldspezifikation die Länge und/oder die Ausrichtung angeben. Damit die Daten linksbündig ausgegeben werden, geben Sie ein Minuszeichen (-) vor der Längenangabe ein. Ansonsten werden die Daten rechtsbündig ausgegeben.

Für die Standardausgabe des Befehls LIST gilt folgende Spezifikation:

```
%-24N %7L %A %D %T
```

Beispiel 1:

```
> LIST Dirs
Prefs            Dir  ----rwed  27-Jun-93    11:43:59
T                Dir  ----rwed  16-Jul-93    11:37:43
Trashcan         Dir  ----rwed  21-Jun-93    17:54:20
```

Nur die Verzeichnisse im aktuellen Verzeichnis (in diesem Fall SYS:) werden angezeigt (obige Abbildung zeigt eine gekürzte Version der normalen Anzeige).

Beispiel 2:

```
1> LIST LI#? TO RAM:Libs.Datei
```

Alle Angaben zu Dateien und Verzeichnissen, die mit "LI" beginnen, werden in der Datei "Libs.Datei" im RAM: gespeichert.

Beispiel 3:

```
1> LIST DF0:Dokumente UPTO 09-Okt-93
```

Nur die Dateien und Verzeichnisse im Verzeichnis "Dokumente" in Laufwerk DF0:, die seit dem 9. Oktober 1993 nicht geändert wurden, werden angezeigt.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls LIST.

LOADRESOURCE

Mit diesem Befehl werden Ressourcen in den Speicher geladen, um Disk-Wechsel auf ein Minimum zu reduzieren.

Format	LOADRESOURCE {<Name>} [LOCK|UNLOCK]
Schablone	NAME/M,LOCK/S,UNLOCK/S
Pfad	C:

Über LOADRESOURCE wird die Anzahl der erforderlichen Diskettenwechsel in Diskettensystemen reduziert, indem folgende Ressourcen in den Speicher geladen werden:

Bibliotheken	Geben Sie den Pfadnamen zur Bibliothek (Library) an.
Geräte	Geben Sie den Pfadnamen zum Gerät an. Geräte können nicht speicherresident (Option LOCK) geladen werden.
Zeichen-sätze	Geben Sie den exakten Pfadnamen der zu ladenden Zeichensatzdatei (Font) ein.
Kataloge	Geben Sie den Pfadnamen wie folgt ein: LOCALE:Catalogs/<Sprache>/Sys/<Katalog>.

Über die Option {<Name>} werden die Pfade zu den zu ladenden Ressourcen angegeben. Die Option LOCK gibt an, daß Ressourcen, z. B. Bibliotheken, Zeichensätze und Kataloge, speicherresident geladen werden. Dadurch wird verhindert, daß die Ressource bei geringer freier Speicherkapazität aus dem RAM ausgelagert wird. Geräte können zwar mit dem Befehl LOADRESOURCE in den Speicher geladen werden, aber Sie können nicht über die Option LOCK angeben, daß Geräte speicherresident sein sollen. Über die Option UNLOCK wird dem Befehl mitgeteilt, daß die Ressource nicht mehr speicherresident geladen bleiben soll, so daß sie aus dem RAM ausgelagert werden kann.

Bei Eingabe von LOADRESOURCE ohne Optionen werden alle speicherresidenten Ressourcen (Option LOCK) im RAM angezeigt.

Beispiel 1:

```
LOADRESOURCE LIBS:asl.library
```

Mit diesem Befehl wird die Bibliothek "asl.library" in den Speicher geladen. Wenn Sie in der Befehlszeile nicht die Option LOCK eingeben, kann diese Bibliothek aus dem RAM ausgelagert werden, wenn nicht mehr genügend freier Speicherplatz zur Verfügung steht.

Beispiel 2:

```
LOADRESOURCE FONTS:topaz/11
```

Mit diesem Befehl wird der Zeichensatz "Topaz 11" in den Speicher geladen.

Beispiel 3:

```
LOADRESOURCE LOCALE:Catalogs/deutsch/Sys/
monitors.catalog
```

Dies ist ein gültiger Pfadname.

LOADWB

Mit diesem Befehl wird die Workbench gestartet.

Format	LOADWB [-DEBUG] [DELAY] [CLEANUP] [NEWPATH]
Schablone	-DEBUG/S,DELAY/S,CLEANUP/S,NEWPATH/S
Pfad	C:

Mit LOADWB wird die Workbench gestartet. Normalerweise geschieht dies bereits beim Systemstart, wenn LOADWB in der Datei "Startup-sequence" aufgerufen wird. Wenn Sie die Workbench schließen, können Sie sie mit dem Befehl LOADWB von einer Shell aus wieder starten.

Mit der Option -DEBUG wird speziell für Entwickler das Menü "Debug" verfügbar, das über die Menüleiste der Workbench ausgewählt werden kann. Wenn die Option DELAY angegeben wird, wartet LOADWB drei Sekunden bis zur Ausführung, damit der Zugriff auf

die Disk vorher beendet wird. Mit der Option CLEANUP wird das anfängliche Disk-Fenster automatisch "aufgeräumt".

Die Workbench merkt sich die beim Ausführen des Befehls LOADWB aktuellen Pfade. Diese werden für die von der Workbench aus gestarteten Shells verwendet. Mit NEWPATH können Sie einen neuen Pfad anwählen, der von der aktuellen Shell übernommen wird.

Beispiel 1:

Wenn Sie die Workbench verlassen haben und über eine Shell arbeiten, geben Sie folgenden Befehl ein:

```
1> LOADWB
```

Damit wird die Workbench wieder aufgerufen. Wenn Sie LOADWB eingeben, obwohl die Workbench bereits geladen ist, wird der Befehl ignoriert.

Beispiel 2:

```
1> PATH DF2:bin ADD
1> LOADWB NEWPATH
```

Mit diesem Befehl wird die Workbench geladen. Alle Shells, die dann über ihr Piktogramm gestartet werden, haben den gleichen Pfad wie die Shell, von der aus der Befehl LOADWB NEWPATH aufgerufen wird.

LOCK

Mit diesem Befehl wird der Schreibschutz für ein Gerät aktiviert bzw. inaktiviert.

Format	LOCK <Laufwerk> [ON | OFF] [<Paßwort>]
Schablone	DRIVE/A,ON/S,OFF/S,PASSKEY
Pfad	C:

Mit LOCK wird der Schreibschutz für ein Gerät bzw. eine Partition aktiviert bzw. inaktiviert. Der Status wird so lange beibehalten, bis das System neu gestartet oder der Befehl LOCK OFF ausgeführt wird.

Sie können auch ein Paßwort angeben. Wenn beim Aktivieren des Schreibschutzes einer Festplattenpartition ein Paßwort benutzt wurde, muß dieses auch zum Aufheben des Schreibschutzes angegeben werden. Das Paßwort kann beliebig lang sein.

Beispiel:

```
1> LOCK Arbeit: ON Geheimcode
```

Die Partition "Arbeit:" ist nun schreibgeschützt. Sie können den Inhalt von "Arbeit:" zwar mit Befehlen wie DIR, LIST oder MORE lesen, Änderungen sind jedoch nicht möglich. Wenn Sie versuchen, eine Datei im Verzeichnis "Arbeit:" zu bearbeiten, erscheint ein Dialogfenster mit dem Hinweis, daß "Arbeit:" schreibgeschützt ist. Sie geben z. B. folgenden Befehl zum Erstellen eines neuen Verzeichnisses ein:

```
1> MAKEDIR Arbeit:Test
```

Daraufhin erscheint folgende Nachricht:

```
Can't create directory Arbeit:Test (Verzeichnis Ar-
beit:Test kann nicht erstellt werden)
Disk ist schreibgeschützt
```

Wenn der Schreibschutz der Partition aufgehoben werden soll, geben Sie folgenden Befehl ein:

```
1> LOCK Arbeit: OFF Geheimcode
```

Ein Gerät kann jeweils nur für die aktuelle Sitzung schreibgeschützt werden. Wenn der Amiga neu gestartet oder ausgeschaltet wird, wird der Schutz aufgehoben.

MAGTAPE

Mit diesem Befehl können SCSI-Bänder (SCSI - Small Computer System Interface) nachgespannt, zurück- oder vorgespult werden.

Format	MAGTAPE [*DEVICE* <Gerätename>] [*UNIT* <n>] [RET | RETENSION] [REW | REWIND] [*SKIP* <n>]
Schablone	DEVICE/K,UNIT/N/K,RET=RETENSION/S, REW=REWIND/S,SKIP/N/K
Pfad	C:

Standardmäßig wird mit diesem Befehl SCSI.device UNIT 4 (Einheit 4) angesprochen. Zum Ändern der Vorgabe müssen Sie sowohl das Schlüsselwort DEVICE als auch UNIT angeben.

Mit der Option RET|RETENSION wird das Band bis zum Ende durchgespult und dann zurückgespult. Mit der Option REW|REWIND wird das Band zurückgespult. Mit der Option SKIP <n> können Sie <n> Dateien auf dem Band überspringen.

Bei Eingabe des Befehls MAGTAPE wird überprüft, ob die Einheit bereit ist, bevor der Befehl gesendet wird. Wenn das Magnetband nicht online ist, müssen Sie den Befehl MAGTAPE wiederholen.

Beispiel:

```
1> MAGTAPE DEVICE second_scsi.device UNIT 0 REW
```

MAKEDIR

Mit diesem Befehl wird ein neues Verzeichnis angelegt.

Format	MAKEDIR {<Name>}
Schablone	NAME/M
Pfad	C:

Mit dem Befehl MAKEDIR werden neue, leere Verzeichnisse mit den angegebenen Namen angelegt. Mit diesem Befehl werden die Verzeichnisse nur in einer Verzeichnisebene erstellt. Alle im Pfad angegebenen Verzeichnisse müssen deshalb schon vorhanden sein. Der Befehl wird abgebrochen, wenn sich im zugehörigen übergeordneten Verzeichnis schon eine Datei oder ein Verzeichnis mit demselben Namen befindet.

Mit dem Befehl MAKEDIR wird für das neue Verzeichnis kein Schubladen-Piktogramm erstellt.

Beispiel 1:

```
1> MAKEDIR Tests
```

Mit diesem Befehl wird im aktuellen Verzeichnis das Verzeichnis "Tests" angelegt.

Beispiel 2:

```
1> MAKEDIR DF1:Xyz
```

Mit diesem Befehl wird im Hauptverzeichnis der Disk in Laufwerk DF1: das Verzeichnis "Xyz" angelegt.

Beispiel 3:

```
1> CD DF0:
1> MAKEDIR Dokumente Verbindlichkeiten Aufträge
```

Mit diesen Befehlen werden auf der Disk in Laufwerk DF0: die drei Verzeichnisse "Dokumente", "Verbindlichkeiten" und "Aufträge" angelegt.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls MAKEDIR.

MAKELINK

Mit diesem Befehl werden Verbindungen zwischen Dateien angegeben.

Format	MAKELINK [FROM] <Datei> [TO] <Datei> [HARD] [FORCE]
Schablone	FROM/A,TO/A,HARD/S,FORCE/S
Pfad	C:

Mit dem Befehl MAKELINK wird auf einer Disk eine FROM-Datei angelegt, die als Verweis (Verbindung, engl. Link) auf eine andere Datei, die TO-Datei, dient. Wenn eine Anwendung oder ein Befehl auf die FROM-Datei zugreift, wird tatsächlich die TO-Datei benutzt. Standardmäßig werden sogenannte "feste" Verbindungen (hard links) unterstützt, d. h. die mit FROM und die mit TO angegebene Datei müssen sich auf demselben Datenträger befinden.

Sogenannte "lose" Verbindungen (soft links), d. h. Verbindungen über verschiedene Datenträger hinweg, sind noch nicht möglich.

Normalerweise unterstützt MAKELINK keine Verbindungen zwischen Verzeichnissen. Um doch eine Verzeichnisverbindung zu erstellen, müssen Sie die Option FORCE angeben. Wenn MAKELINK feststellt, daß Sie eine Kreisverbindung erstellt haben, z. B. eine Verbindung zum Mutterverzeichnis, wird die Meldung **"Link loop not allowed"** (Verbindungsschleife nicht zulässig) angezeigt.

MOUNT

Mit diesem Befehl wird ein an das System angeschlossenes Gerät verfügbar gemacht.

Format	MOUNT {Gerät} [*FROM* <Dateiname>]
Schablone	DEVICE/M,FROM/K
Pfad	C:

Vom Befehl MOUNT werden die Gerätekonfigurationsparameter aus einer Datei gelesen. Anschließend werden diese Parameter zum Anmelden des Geräts bzw. Bereitstellen des Geräts im System verwendet. Über einen Befehl können mehrere Geräte angemeldet werden. Mit dem Argument {Gerät} werden die Namen der anzumeldenen Geräte angegeben.

Vom Befehl MOUNT können sowohl DOSDrivers-Anmeldedateien als auch herkömmliche MountList-Dateien mit mehreren Einträgen verarbeitet werden. Die Verarbeitungsmethode hängt dabei davon ab, mit welcher der folgenden drei Methoden die Argumente angegeben werden:

1. Bei Angabe eines Gerätenamens wird nacheinander in den Verzeichnissen "DEVS:DOSDrivers" und "SYS:Storage/DOSDrivers" eine Anmeldedatei mit diesem Namen gesucht. Ist keine Anmeldedatei dieses Namens vorhanden, wird in der Datei "DEVS:MountList" nach einem entsprechenden Eintrag gesucht. Dies ist die beste Methode, wenn auf Ihrem System nur eine Konfiguration für das jeweilige Gerät vorhanden ist.

2. Bei Angabe eines Pfads wird im angegebenen Verzeichnis nach der entsprechenden Anmeldedatei gesucht. Zum Anmelden von mehreren Geräten können Jokerzeichen verwendet werden (z. B. **MOUNT DEVS:DOSDrivers/~(#?.info)**). Verwenden Sie

diese Methode, wenn die Anmeldedateien nicht in den DOSDrivers-Verzeichnissen (bzw. DOSDrivers-Schubladen) gespeichert sind oder gleichzeitig mehrere Anmeldedateien verarbeitet werden sollen.

3. Bei Angabe des Schlüsselworts FROM und eines Pfads wird über den Befehl MOUNT der genaue Standort der zu verarbeitenden MountList-Datei angegeben. Verwenden Sie diese Methode, wenn eine MountList-Datei nicht im Verzeichnis DEVS: gespeichert ist oder mehrere MountList-Dateien vorhanden sind.

Hinweis Die Merkmale des Piktogramms für eine Anmeldedatei setzen ggf. die Parameter außer Kraft, die in der Anmeldedatei selbst angegeben sind.

Beispiel 1:

```
1> MOUNT PIPE:
```

Bei diesem Befehl wird die Datei "DEVS:DOSDrivers/PIPE" gesucht und ggf. verarbeitet. Wenn die Datei "DEVS:DOSDrivers/PIPE" nicht vorhanden ist, wird nach der Datei "SYS:Storage /DOSDrivers/PIPE" gesucht. Ist diese ebenfalls nicht vorhanden, wird in der Datei "DEVS:MountList" nach dem Eintrag "PIPE:" gesucht.

Beispiel 2:

```
1> MOUNT Arbeit:Geräte/PIPE
```

Bei diesem Befehl wird im Verzeichnis "Arbeit:Geräte" nach der Anmeldedatei PIPE gesucht.

Beispiel 3:

```
1> MOUNT PIPE: FROM SYS:MeinGerät/MountList
```

Bei diesem Befehl wird in der Datei "SYS:MeinGerät/MountList" nach dem Eintrag PIPE: gesucht.

Anhang B enthält weitere Informationen zu MountList-Dateien.

NEWCLI

Mit diesem Befehl wird ein neues Shell-Fenster geöffnet.

Format	NEWCLI [<Fensterspezifikation>] [FROM <Dateiname>]
Schablone	WINDOW,FROM
Pfad	Intern

Mit dem Befehl NEWCLI wird wie mit dem Befehl NEWSHELL ein neuer Shell-Prozeß gestartet (siehe dort).

NEWSHELL

Mit diesem Befehl wird ein neues Shell-Fenster geöffnet.

Format	NEWSHELL [<Fensterspezifikation>] [FROM <Dateiname>]
Schablone	WINDOW,FROM
Pfad	Intern

Das neue Fenster wird automatisch das aktive Fenster und wird zum aktiven Prozeß. Das aktuelle Verzeichnis, die Eingabeaufforderung, der Pfad, die lokalen Umgebungsvariablen und die Stack-Größe stimmen mit dem Ausgangsfenster überein. Die Shell-Fenster sind aber dennoch voneinander unabhängig und ermöglichen separate Ein-/Ausgabe sowie Programmausführung.

Wie die meisten anderen Amiga-Fenster kann dieses Fenster vergrößert, verkleinert, verschoben, gezoomt und vor oder hinter anderen Fenstern plaziert werden.

Soll ein individuell angepaßtes Fenster geöffnet werden, verwenden Sie das Argument WINDOW. Sie können dann bei <Fensterspezifikation> die Ausgangsgröße, die Position und den Titel des Fenster angeben. Die Syntax lautet wie folgt:

```
CON:x/y/Breite/Höhe/Titel/Optionen
```

Dabei gilt:

x	Anzahl der Bildpunkte vom linken Bildschirmrand zum linken Rand des Shell-Fensters. Wenn Sie keinen Wert (//) angeben, wird die niedrigstmögliche Anzahl von Bildpunkten verwendet.
y	Anzahl der Bildpunkte vom oberen Bildschirmrand zum oberen Rand des Shell-Fensters. Wenn Sie keinen Wert (//) angeben, wird die niedrigstmögliche Anzahl von Bildpunkten verwendet.
Breite	Breite des Shell-Fensters in Bildpunkten. Wenn Sie keinen Wert (//) angeben, wird die niedrigstmögliche Anzahl von Bildpunkten verwendet.
Höhe	Höhe des Shell-Fensters in Bildpunkten. Wenn Sie keinen Wert (//) angeben, wird die niedrigstmögliche Anzahl von Bildpunkten verwendet.
Titel	Text in der Titelleiste des Shell-Fensters.

Trennen Sie die Parameter und Optionen durch Schrägstriche voneinander. Falls die Fensterspezifikation Leerzeichen enthält, müssen Sie das gesamte Argument in doppelte Anführungszeichen setzen (").

Folgende Optionen sind zulässig:

AUTO	Das Fenster wird automatisch angezeigt, wenn für das Programm Eingabedaten erforderlich sind oder Daten ausgegeben werden. Nach Öffnen des Shell-Fensters können sofort Daten eingegeben werden. Das Fenster kann nur mit dem Befehl ENDSHELL geschlossen werden. Bei Auswahl des Schließsymbols wird das Fenster zwar geschlossen, aber sofort wieder geöffnet, da Eingaben erwartet werden.
ALT	Wenn Sie das Zoom-Symbol anklicken, wird das Fenster in der angegebenen Position mit der angegebenen Größe angezeigt. Die vier Parameter müssen durch Schrägstriche voneinander getrennt sein (z. B. ALT30/30/200/200).
BACKDROP	Das Fenster wird im Hintergrund hinter allen Workbench-Fenstern angezeigt. Dieses Shell-Fenster kann nicht in den Vordergrund geholt werden. Sie müssen die Größe der anderen Workbench-Fenster ändern, damit sie es sehen können.

CLOSE Das Fenster verfügt über alle Standardsymbole einschließlich des Schließsymbols. Dabei handelt es sich zwar um die Vorgabe für Shell-Fenster, aber wenn Sie das Argument WINDOW verwenden, müssen Sie diese Option zum Aufrufen eines Standard-Shell-Fensters angeben.

INACTIVE Das Fenster wird zwar geöffnet, aber nicht zum aktiven Fenster gemacht.

NOBORDER Das Fenster wird ohne linken und unteren Rand geöffnet. Es stehen nur das Zoom-Symbol, das Vorder-/Hintergrundsymbol und das Größensymbol zur Verfügung.

NOCLOSE Im Fenster wird kein Schließsymbol angezeigt. Wenn Sie ein Konsolenfenster auf normalem Weg öffnen, ist kein Schließsymbol vorhanden. Wenn Sie jedoch ein Konsolenfenster unter Angabe der Option AUTO öffnen, wird automatisch ein Schließsymbol im Fenster angezeigt.

NODEPTH Im Fenster wird kein Vorder-/Hintergrundsymbol angezeigt.

NODRAG Das Fenster kann nicht verschoben werden. Es verfügt zwar über das Zoom-Symbol und das Vorder-/Hintergrundsymbol, hat aber kein Schließsymbol.

NOSIZE Das Fenster verfügt nur über ein Vorder-/Hintergrundsymbol.

SCREEN Das Fenster wird auf einem Public-Schirm geöffnet. Dieser Public-Schirm muß dabei bereits vorhanden sein. Hinter dem Schlüsselwort SCREEN müssen Sie den Namen des Public-Schirms angeben.

SIMPLE Wenn Sie das Fenster vergrößern, füllt der Text den neuen verfügbaren Bereich auf. Sie können jetzt Text sehen, der aus dem Anzeigebereich gerollt war. Dies ist die Vorgabe für Standard-Shells.

SMART Wenn Sie das Fenster vergrößern, füllt der Text nicht den neuen verfügbaren Bereich auf. Dadurch wird weniger Speicherplatz belegt.

WAIT Das Fenster kann nur durch Auswählen des Schließsymbols oder durch Drücken der Tastenkombination Ctrl-\ geschlossen werden. Ist WAIT die einzige Option, wird kein Schließsymbol angezeigt.

Von NEWSHELL wird normalerweise die Vorgabestartdatei "S:Shell-startup" verwendet, wenn keine andere Datei mit FROM angegeben wurde. Bei der Datei "S:Shell-startup" handelt es sich um eine vorgegebene AmigaDOS-Befehlsdatei. Sie können mehrere

Startdateien verwenden, die jeweils verschiedene Alias-Namen für Befehle enthalten. Diese individuell angepaßten Shell-Umgebungen können Sie mit FROM aufrufen.

Mit diesem Befehl wird wie mit dem Befehl NEWCLI ein neuer Shell-Prozeß gestartet.

Beispiel 1:

```
1> NEWSHELL
```

Mit diesem Befehl wird ein neues Shell-Fenster mit der vorgegebenen Fensterspezifikation geöffnet.

Beispiel 2:

```
1> NEWSHELL "CON://640/200/Meine Shell/CLOSE"
```

Mit diesem Befehl wird ein Fenster geöffnet, dessen linke obere Ecke in der linken oberen Ecke des Bildschirms liegt ist. Das Fenster ist 640 Bildpunkte breit und 200 Bildpunkte hoch. Das Fenster hat den Titel "Meine Shell" und verfügt über ein Schließsymbol. Das gesamte Argument wurde in Anführungszeichen gesetzt, da der Fenstertitel ein Leerzeichen enthält. Wenn Sie diesen Befehl in die Datei "User-startup" einfügen, wird dieses Shell-Fenster automatisch bei jedem Systemstart geöffnet.

Beispiel 3:

```
1> NEWSHELL FROM S:Programming.startup
```

Mit diesem Befehl wird eine neue Shell geöffnet. Es wird aber nicht die Datei "Shell-startup", sondern die Datei "Programming.startup" ausgeführt. Sie können in dieser Startdatei z. B. Alias-Namen und Eingabeaufforderungsbefehle hinzufügen, die Sie nur beim Programmieren benutzen.

Kapitel 8 enthält weitere Beispiele nur Verwendung des Befehls NEWSHELL.

PATH

Mit diesem Befehl wird die Liste der Verzeichnisse verwaltet, in denen die Shell Befehle sucht.

Format	PATH [{<dir>}] [ADD] [SHOW] [RESET] [REMOVE] [QUIET]
Schablone	PATH/M,ADD/S,SHOW/S,RESET/S,REMOVE/S, QUIET/S,
Pfad	Intern

Mit dem Befehl PATH können Sie sich den Suchpfad von AmigaDOS anzeigen lassen, Elemente dem Pfad hinzufügen und den Pfad ändern. Dieser Suchpfad wird von AmigaDOS bei der Suche nach einem auszuführenden Befehl bzw. Programm verwendet. Wenn ein Befehl in einem Verzeichnis des Suchpfads liegt, müssen Sie nicht den vollständigen Pfad zu diesem Befehl eingeben. Sie geben einfach den Befehlsnamen ein, und AmigaDOS sucht in allen Verzeichnissen im Suchpfad nach der zugehörigen Datei.

Hinweis — Der Suchpfad ist nur relevant, wenn von AmigaDOS nach einem auszuführenden Befehl bzw. Programm gesucht wird. In Argumenten für Befehle wie z. B. COPY und DELETE müssen Sie stets den vollständigen Pfad angeben.

Wenn Sie den Befehl PATH ohne Argumente oder mit der Option SHOW verwenden, werden die Verzeichnisse im aktuellen Suchpfad angezeigt. Wenn ein im Suchpfad angegebener Datenträger nicht gefunden wird, erscheint normalerweise ein Dialogfenster. Wenn z. B. eine Diskette zum Suchpfad gehört und sich diese Diskette nicht im Laufwerk befindet, werden Sie im Dialogfenster aufgefordert, diese Diskette einzulegen.

Bei Angabe der Option QUIET wird kein Dialogfenster angezeigt, wenn ein nicht angemeldeter Datenträger gefunden wird. In diesem Fall wird die Meldung **`Gerät (oder Datenträger) ist nicht angemeldet`** ausgegeben. Die Namen der Verzeichnisse auf einem solchen Datenträger werden nicht angezeigt.

Mit der Option ADD können Sie den Namen von Verzeichnissen angeben, die zusätzlich in den aktuellen Suchpfad aufnommen werden sollen. Mit dem Befehl PATH ADD können Sie gleichzeitig eine beliebige Anzahl von Verzeichnissen dem Suchpfad hinzufügen (das Schlüsselwort ADD muß dabei nicht angegeben werden). Die Namen der einzelnen Verzeichnisse müssen durch mindestens ein Leerzeichen voneinander getrennt werden. Wenn der Befehl PATH ausgeführt wird, sucht AmigaDOS alle mit ADD neu hinzugefügten Verzeichnisse.

Wenn Sie den bestehenden Suchpfad vollständig durch einen neuen ersetzen wollen, geben Sie PATH RESET gefolgt von den Namen der neuen Verzeichnisse ein. Der bestehende Suchpfad wird mit Ausnahme des aktuellen Verzeichnisses und von Verzeichnis C: gelöscht und durch den neuen ersetzt.

Mit der Option REMOVE wird das angegebene Verzeichnis aus dem Suchpfad gelöscht.

Beispiel:

```
1> PATH EXTRAS:Tools ADD
```

Mit diesem Befehl wird das Verzeichnis "Tools" auf der Diskette "EXTRAS:" in den Suchpfad der Shell eingefügt. Wenn sich die Diskette "EXTRAS:" in keinem Laufwerk befindet, werden Sie durch ein Dialogfenster aufgefordert, die Diskette einzulegen.

Wenn Sie die Diskette "EXTRAS:" aus dem Laufwerk nehmen und den folgenden Befehl eingeben, werden die Verzeichnisse im aktuellen Suchpfad angezeigt. Sie werden durch ein Dialogfenster aufgefordert, die Diskette "EXTRAS:" einzulegen:

```
1> PATH
```

Wenn Sie dagegen den folgenden Befehl eingeben, werden zwar auch die Verzeichnisse im aktuellen Suchpfad angezeigt, aber wenn der Pfad auf "Extras:Tools" trifft, erscheint in der Liste die oben genannte Fehlermeldung:

```
1> PATH QUIET
```

Siehe auch: ASSIGN. Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls PATH.

PROMPT

Mit diesem Befehl wird der Eingabeaufforderungstext der aktuellen Shell geändert.

Format	PROMPT [<Eingabeaufforderung>]
Schablone	PROMPT
Pfad	Intern

Mit diesem Befehl können Sie die Zeichenkette für die Eingabeaufforderung individuell anpassen. Die Eingabeaufforderung ist der Text, der am Anfang einer Befehlszeile angezeigt wird. Die Eingabeaufforderung kann beliebige Zeichen enthalten, einschließlich Escape-Sequenzen.

In den Beispielen dieses Handbuchs wird die Eingabeaufforderung als "1>" dargestellt.

Für die Eingabeaufforderung gilt folgende Vorgabe:

```
"%N.%S> "
```

Dadurch wird die Shell-Nummer, ein Punkt, das aktuelle Verzeichnis, eine schließende spitze Klammer und ein Leerzeichen angezeigt. Wenn Sie den Befehl PROMPT ohne <Eingabeaufforderung> eingeben, wird die Eingabeaufforderung auf diese Vorgabe zurückgesetzt.

Im Argument <Eingabeaufforderung> können folgende Platzhalter verwendet werden:

%N	Die Shell-Nummer wird angezeigt.
%S	Das aktuelle Verzeichnis wird angezeigt.
%R	Der Rückgabecode der letzten Operation wird angezeigt.

Am Ende der Zeichenkette wird nicht automatisch ein Leerzeichen eingefügt. Wenn zwischen der Eingabeaufforderung und der darauffolgenden Eingabe ein Leerzeichen stehen soll, geben Sie das Leerzeichen in die Zeichenkette ein und setzen Sie sie in doppelte Anführungszeichen.

Befehle werden in die Eingabeaufforderung integriert, indem sie zwischen umgekehrte Apostrophe (`) gesetzt werden. Solche Befehle werden allerdings nur einmal bei Aufruf des PROMPT-Befehls ausgeführt und nicht jedesmal neu, wenn die Eingabeaufforderung wieder angezeigt wird

Beispiel 1:

```
1> PROMPT %N
1
```

Nur die Shell-Nummer wird angezeigt, aber nicht mehr die schließende spitze Klammer.

Beispiel 2:

```
1> PROMPT "%N.%S.%R> "
1.Arbeit:Anim.0>
```

Die Shell-Nummer, das aktuelle Verzeichnis und der Rückgabecode des letzten Befehls werden angezeigt. Hinter der schließenden spitzen Klammer steht ein Leerzeichen.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls PROMPT.

PROTECT

Mit diesem Befehl werden die Schutzbits einer Datei bzw. eines Verzeichnisses geändert.

Format	PROTECT [FILE] <Datei | Namensmuster> [FLAGS][+ | -] [<Kennungen>] [ADD | SUB] [ALL] [QUIET]
Schablone	FILE/A,FLAGS,ADD/S,SUB/S,ALL/S,QUIET/S
Pfad	C:

Zu allen Dateien und Verzeichnissen werden Schutzbits gespeichert, mit denen verschiedene Attribute (Datei- und Zugriffsart) der Dateien festgelegt werden. Dateiart und Zugriffsberechtigungen können durch Änderung dieser Bits geändert werden. Der Befehl

PROTECT dient zum Setzen und Löschen von Schutzbits. Bei Verzeichnissen ist nur das Schutzbit "d" von Bedeutung.

Die einzelnen Schutzbits werden als Buchstaben dargestellt:

s	Die Datei ist eine Befehlsdatei (Skript).
p	Die Datei ist ein purer Befehl und kann resident gemacht werden.
a	Die Datei wurde archiviert.
r	Die Datei kann gelesen werden.
w	Die Datei kann geändert werden (Schreibzugriff).
e	Die Datei kann ausgeführt werden (Programm).
d	Die Datei bzw. das Verzeichnis kann gelöscht werden. (Dateien innerhalb eines löschgeschützten Verzeichnisses können dennoch gelöscht werden.)

Mit dem Befehl LIST können Sie sich die Schutzbits einer Datei anzeigen lassen. Die aktivierten Schutzbits werden als Buchstaben angezeigt, die inaktivierten als Bindestriche. Wenn eine Datei z. B. lesbar, schreibbar und löschbar ist, erscheint folgendes: ----rw-d.

Wenn Sie alle Schutzbits gleichzeitig festlegen wollen, geben Sie alle gewünschten Schutzbitbuchstaben als FLAGS-Argument ein (ohne dabei weitere Schlüsselwörter zu verwenden). Die angegebenen Bits werden aktiviert, alle anderen inaktiviert.

Die Zeichen + und - (bzw. die entsprechenden Schlüsselwörter ADD und SUB) werden verwendet, um bestimmte Bits zu aktivieren bzw. zu inaktiveren, ohne den Status der nicht angegebenen Bits zu ändern. Auf + bzw. - folgen die Buchstaben der Bits, die aktiviert bzw. inaktiviert werden sollen. Nur diese Bits werden geändert. Hinter dem Zeichen und zwischen den Buchstaben dürfen keine Leerzeichen stehen. Die Reihenfolge der Buchstaben ist unerheblich. ADD und SUB funktionieren entsprechend, aber zwischen dem Schlüsselwort und den Buchstaben muß ein Leerzeichen eingegeben werden. Sie können mit ein und demselben Befehl Bits nicht gleichzeitig aktivieren und inaktivieren.

Mit der Option ALL können Sie die angegebenen Schutzbits für alle Dateien und Unterverzeichnisse im angegebenen Verzeichnis aktivieren bzw. inaktivieren, deren Namen mit dem Namensmuster übereinstimmen. Mit der Option QUIET wird die Bildschirmausgabe unterdrückt.

Beispiel 1:

```
1> PROTECT DF0:Memo +rw
```

Mit diesem Befehl werden die Schutzbits r (lesbar) und w (schreibbar) für die Datei "Memo" in Laufwerk DF0: aktiviert. Alle anderen Schutzbits bleiben unverändert.

Beispiel 2:

```
1> PROTECT L:#? e SUB
```

Mit diesem Befehl wird das Schutzbit e (ausführbar) bei allen Dateien im Verzeichnis L: inaktiviert.

Beispiel 3:

```
1> PROTECT Arbeit:Bild rwed
```

Mit diesem Befehl erhält die Datei "Arbeit:Bild" den Schutzstatus "----rwed".

QUIT

Mit diesem Befehl wird eine Befehlsdatei mit dem angegebenen Rückgabecode verlassen.

Format	QUIT [<Rückgabecode>]
Schablone	RC/N
Pfad	Intern

Mit QUIT wird die Ausführung einer Befehlsdatei beim angegebenen Rückgabecode abgebrochen. Der Standardrückgabecode ist 0. Es wird empfohlen, die Standardrückgabecodes 5, 10 und 20 zu verwenden.

Beispiel:

```
ASK "Wollen Sie jetzt abbrechen?"
IF WARN
   QUIT 5
ENDIF
ECHO "OK"
ECHO "Die Befehlsdatei wird fortgesetzt."
```

Wenn Sie hinter der Eingabeaufforderung "Y" (für Yes = Ja) eingeben, wird die Befehlsdatei abgebrochen, da WARN dem Rückgabecode 5 entspricht. Wenn Sie "N" (Nein) oder die Eingabetaste drücken, erscheint folgendes im Shell-Fenster:

```
OK
Die Befehlsdatei wird fortgesetzt.
```

RELABEL

Mit diesem Befehl wird der Datenträgername der Disk im angegebenen Laufwerk in den angegebenen Namen geändert.

Format	RELABEL [DRIVE] <Laufwerk> [NAME] <Name>
Schablone	DRIVE/A,NAME/A
Pfad	C:

Datenträgernamen werden erstmals beim Formatieren einer Disk festgelegt. Mit dem Befehl RELABEL können Sie den Datenträgernamen der Disk beliebig ändern.

Bei Diskettensystemen mit nur einem Laufwerk müssen Sie dabei immer über den Datenträgernamen und nicht über den Laufwerksnamen auf Disketten zugreifen.

Beispiele:

```
1> RELABEL Workbench: MeineDisk
```

Mit diesem Befehl wird die Workbench-Disk in "MeineDisk" umbenannt. Beachten Sie, daß nach dem zweiten Namen kein Doppelpunkt eingegeben werden muß.

```
1> RELABEL DF2: DatenDisk
```

Mit diesem Befehl wird der Name der Disk in DF2: in "DatenDisk" geändert.

REMRAD

Mit diesem Befehl wird die resetfeste RAM-Disk entfernt.

Format	REMRAD [<Gerät>] [FORCE]
Schablone	DEVICE,FORCE/S
Pfad	C:

Wenn Sie die resetfeste RAM-Disk (normalerweise als RAD: angemeldet) löschen und dazu das System nicht ausschalten wollen, verwenden Sie den Befehl REMRAD. Wenn mehr als eine resetfeste RAM-Disk angemeldet ist, muß auch das <Gerät> (DEVICE) angegeben werden.

Über REMRAD wird RAD: inaktiviert und alle zugehörigen Dateien werden gelöscht. Das Disk-Piktogramm "RAD: RAM_0" wird jedoch weiterhin angezeigt. Beim nächsten Neustart des Amiga wird RAD: vollständig aus dem Speicher gelöscht, und das Piktogramm wird nicht mehr angezeigt.

Ist die RAM-Disk beim Aufrufen des Befehls REMRAD gerade in Verwendung, wird der Befehl mit der Meldung "device in use" (Gerät in Verwendung) abgebrochen. Soll in diesem Fall die RAM-Disk dennoch entfernt werden, müssen Sie die Option FORCE verwenden.

RENAME

Mit diesem Befehl wird der Name bzw. der Standort einer Datei bzw. eines Verzeichnisses geändert.

Format	RENAME [FROM] {<Name>} [TO|AS] <Name>
Schablone	FROM/A/M,TO=AS/A,QUIET/S
Pfad	C:

Mit dem Befehl RENAME wird die mit FROM angegebene Datei bzw. das Verzeichnis in den mit TO angegebenen Namen umbenannt.

Dabei müssen sich die Angaben bei FROM und TO auf denselben Datenträger beziehen. Bei der Umbenennung eines Verzeichnisses bleibt der Inhalt dieses Verzeichnisses unverändert (die Verzeichnisse und Dateien innerhalb dieses Verzeichnisses behalten ihren Namen und Inhalt bei). Wenn als FROM-Argument mehrere Elemente angegeben werden, muß ein Verzeichnis als TO-Argument angegeben werden.

Wenn Sie ein Verzeichnis umbenennen oder mit RENAME eine Datei in ein anderes Verzeichnis verlegen, ändert AmigaDOS auch die Position des Verzeichnisses bzw. der Datei in der Verzeichnishierarchie. Dies ist eine effiziente Methode zum Verlegen von Elementen.

Beispiel 1:

```
1> RENAME Arbeit/Bsp1 AS :Test/Bsp2
```

Mit diesem Befehl wird die Datei "Bsp1" in "Bsp2" umbenannt und vom Verzeichnis "Arbeit" in das Verzeichnis " :Test" verlegt. Damit dieser Befehl ausgeführt werden kann, muß das Verzeichnis "Test" bereits im Hauptverzeichnis vorhanden sein.

Beispiel 2:

```
1> RENAME 3.doc 5.doc a.doc TO Dokumente
```

Mit diesem Befehl werden die Dateien "3.doc", "5.doc" und "a.doc" in das Verzeichnis "Dokumente" verlegt. Dieses Verzeichnis muß bereits vorhanden sein.

REQUESTCHOICE

Mit diesem Befehl wird ermöglicht, daß AmigaDOS- und ARexx-Befehlsdateien benutzerspezifische Dialogfenster verwenden.

Format	REQUESTCHOICE <Titel> <Text> {<Felder>} [*PUBSCREEN* <Public-Schirm>]
Schablone	TITLE/A,BODY/A,GADGETS/A/M, PUBSCREEN/K
Pfad	C:

Mit dem Argument <Titel> wird der Titel des Dialogfensters angegeben.

Mit dem Argument <Text> wird der Text des Dialogfensters angegeben. Zeilenvorschübe werden als *N eingefügt.

Mit dem Argument <Felder> wird der Text für die verschiedenen Felder angegeben. Die einzelnen Felder werden durch Leerzeichen voneinander getrennt.

Die Nummer des ausgewählten Felds wird als Ergebnis an die Konsole ausgegeben. Zur Auswertung in einer Befehlsdatei können die Ausgabedaten in eine Umgebungsvariable umgeleitet werden. Wenn das Dialogfenster nicht geöffnet werden konnte, wird der Rückgabecode 20 generiert.

Bei Angabe des Arguments PUBSCREEN kann das Dialogfenster auf einem Public-Schirm geöffnet werden.

Beispiel:

```
1> RequestChoice >ENV:rcnum "Neuer Titel" "Dies ist
   mein Dialogfenster*nWählen Sie ein Symbol" "OK"
   "Vielleicht" "Abbrechen"
```

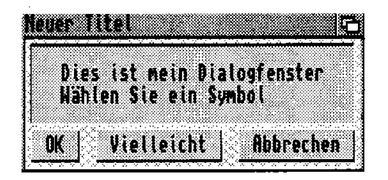


Abb. 6-1. Mit REQUESTCHOICE erstelltes Dialogfenster (Beispiel)

ENV:rcnum enthält nach Auswahl eines Felds den Wert 0, 1 oder 2. Von der Befehlsdatei kann dieser Wert zur Steuerung der weiteren Verarbeitung verwendet werden.

REQUESTFILE

Ermöglichen, daß AmigaDOS- und ARexx-Befehlsdateien Dateiauswahlfenster verwenden.

Format REQUESTFILE [DRAWER <Schubladenname>] [*FILE* <Datei>] [*PATTERN* <Namensmuster>] [*TITLE* <Titel>] [*POSITIVE* <Text>] [*NEGATIVE* <Text>] [*ACCEPTPATTERN* <Namensmuster>] [*REJECTPATTERN* <Namensmuster>] [SAVEMODE] [MULTISELECT] [DRAWERSONLY] [NOICONS] [*PUBSCREEN* <Public-Schirm>]

Schablone DRAWER,FILE/K,PATTERN/K,TITLE/K, POSITIVE/K,NEGATIVE/K, ACCEPTPATTERN/K,REJECTPATTERN/K, SAVEMODE/S,MULTISELECT/S, DRAWERSONLY/S,NOICONS/S,PUBSCREEN/K

Pfad C:

Bei Angabe ohne Argumente wird ein Dateiauswahlfenster mit den Feldern OK, Volumes, Parent und Cancel (OK, Laufw., Mutterv., Abbrechen) erstellt. Die Felder "Schublade" und "Datei" sind leer, und es wird der Inhalt des aktuellen Verzeichnisses angezeigt.

Mit dem Argument DRAWER wird der anfängliche Inhalt des Schubladenfelds angegeben.

Mit dem Argument FILE wird der anfängliche Inhalt des Dateinamenfelds angegeben.

Im Argument PATTERN kann ein standardmäßiges Namensmuster verwendet werden. Der anfängliche Inhalt des dann angezeigten Namensmustersymbols wird angegeben. Wenn diese Option nicht verwendet wird, hat das Dateiauswahlfenster kein Namensmustersymbol.

Mit der Option TITLE wird der Titel des Dateiauswahlfensters angegeben.

Mit der Option POSITIVE wird der Text im linken (positiven) Teil des Dateiauswahlfensters angegeben.

Mit der Option NEGATIVE wird der Text im rechten (negativen) Teil des Dateiauswahlfensters angegeben.

Mit der Option ACCEPTPATTERN wird ein Standardnamensmuster angegeben. Nur die Dateien, deren Namen diesem Muster entsprechen, werden im Dateiauswahlfenster angezeigt.

Mit der Option REJECTPATTERN wird ein Standardnamensmuster angegeben. Dateien, deren Namen diesem Muster entsprechen, werden nicht im Dateiauswahlfenster angezeigt.

Wenn SAVEMODE angegeben ist, dient das Dateiauswahlfenster zum Schreiben von Dateien auf Disk. Bei MULTISELECT können aus dem Fenster mehrere Dateien gleichzeitig ausgewählt werden. Wenn DRAWERSONLY angegeben wird, hat das Dateiauswahlfenster kein Dateinamenfeld. Damit wird das Dateiauswahlfenster praktisch zu einem Verzeichnisauswahlfenster. Bei NOICONS werden im Fenster keine Piktogrammdateien angezeigt (.info-Dateien).

Die ausgewählten Dateien werden zwischen Anführungszeichen und durch Leerzeichen getrennt als Ergebnis in die Befehlszeile gestellt. Wenn eine Datei ausgewählt wurde, wird der Rückgabecode 0 erzeugt, wenn keine Auswahl erfolgte, wird der Rückgabecode 5 ausgegeben.

Das Argument PUBSCREEN ermöglicht das Öffnen des Auswahlfensters auf einem Public-Schirm.

Beispiel:

```
1> REQUESTFILE DRAWER Devs: TITLE "Mein Dialog"
NOICONS
```

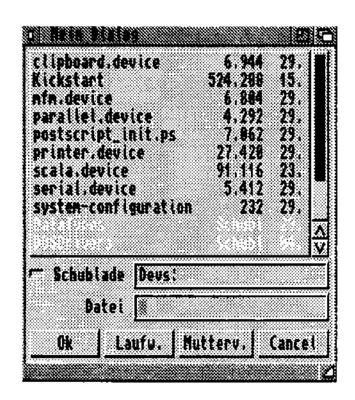


Abbildung 6-2. Beispiel eines RequestFile-Auswahlfensters

RESIDENT

Liste residenter Befehle anzeigen und bearbeiten.

Format	RESIDENT [<residenter Name>] [<Dateiname>] [REMOVE] [ADD] [REPLACE] [PURE | FORCE] [SYSTEM]
Schablone	NAME,FILE,REMOVE/S,ADD/S, REPLACE/S,PURE=FORCE/S,SYSTEM/S
Pfad	Intern

Mit diesem Befehl können Befehle geladen und der Liste residenter Befehle hinzugefügt werden, die durch die Shell verwaltet werden. Dadurch kann der Befehl ausgeführt werden, ohne daß er jedesmal neu geladen werden muß. Wenn RESIDENT ohne Optionen aufgerufen wird, listet der Befehl die Programme auf, die in der Liste residenter Programme stehen.

Nur bestimmte Befehle, die sog. "puren" Befehle, können resident geladen werden. Solche Befehle sind sowohl wiedereintrittsfähig als

auch wiederholt ausführbar. Ein wiedereintrittsfähiger Befehl kann von zwei oder mehr Programmen gleichzeitig und unabhängig voneinander benutzt werden. Ein wiederholt ausführbarer Befehl kann erneut ausgeführt werden, ohne neu geladen werden zu müssen. Befehle, die diese Eigenschaften haben, werden als pure Befehle bezeichnet und haben das Schutzbit p gesetzt.

Die folgenden Befehle können nicht resident geladen werden: BINDDRIVERS, CONCLIP, IPREFS, LOADRESOURCE, LOADWB und SETPATCH.

Wenn Sie wissen wollen, welche Befehle pur sind, können Sie sich mit dem Befehl LIST das Verzeichnis C: anzeigen lassen, in dem die puren Befehle mit dem Schutzbit p gekennzeichnet sind.

Viele der Befehle im Verzeichnis C: und der Befehl MORE im Verzeichnis "Utilities" sind pure Befehle und können resident geladen werden. Wenn bei einem Befehl das Schutzbit p nicht gesetzt ist, kann dieser wahrscheinlich nicht ohne weiteres resident gemacht werden. (Auch wenn das Schutzbit p noch nachträglich gesetzt wird, wird dadurch der Befehl oder das Programm nicht automatisch pur.)

Die Option REPLACE ist die Standardoption und muß nicht ausdrücklich angegeben werden. Wenn kein <residenter Name> angegeben wird (sondern z. B. nur ein Dateiname), wird dieser Dateiname in die Liste residenter Befehle eingetragen. Dabei muß der vollständige Pfad der Datei angegeben werden.

Wenn ein <residenter Name> angegeben wird und ein gleichnamiges Programm bereits in der Liste vorhanden ist, wird versucht, diesen Befehl zu ersetzen (engl. replace). Dieser Name muß dann verwendet werden, um auf die residente Version des Befehls zuzugreifen. Das Ersetzen ist nur dann erfolgreich, wenn der bereits in der Liste vorhandene Befehl gerade nicht verwendet wird.

Wenn Sie die Option REPLACE außer Kraft setzen und mehrere Versionen eines Befehls gleichzeitig resident speichern wollen, verwenden Sie die Option ADD und geben jeder geladenen Version einen anderen <residenten Namen>.

Mit der Option SYSTEM wird der Befehl in den Systemteil der Liste aufgenommen. Die mit dieser Option aufgenommenen Befehle können nicht mehr gelöscht werden. Wenn Sie sich solche Systemdateien anzeigen lassen wollen, verwenden Sie die Option SYSTEM.

Mit der Option PURE werden auch Befehle geladen, die nicht als pur gekennzeichnet sind (Schutzbit p ist nicht gesetzt). Mit dieser Option können Sie testweise andere Befehle und Programme für pur erklären und überprüfen, ob es funktioniert. Bei der Option PURE ist Vorsicht geboten. Stellen Sie sicher, daß die Programme, die Sie RESIDENT laden möchten, die entsprechenden Kriterien erfüllen, und achten Sie darauf, daß Sie diesen Befehl jeweils nur für einen Prozeß gleichzeitig verwenden.

Außerdem kann mit dem Befehl RESIDENT die Verfügbarkeit interner Befehle gesteuert werden. Um einen internen Befehl zu inaktivieren (z. B. für ein Anwendungspaket, das einen eigenen Befehl mit dem gleichen Namen hat), geben Sie folgenden Befehl ein: RESIDENT <Befehl> REMOVE. Der AmigaDOS-Befehl kann mit der Option REPLACE wieder aktiviert werden.

Beispiel 1:

```
1> RESIDENT C:COPY
```

Mit diesem Befehl wird der Befehl COPY resident gemacht (und vorhergehende Versionen werden ersetzt).

Beispiel 2:

```
1> RESIDENT Copy2 DF1:C/COPY ADD
```

Mit diesem Befehl wird eine weitere Version des Befehls COPY unter dem Namen "Copy2" in die Liste residenter Befehle aufgenommen.

Beispiel 3:

```
1> RESIDENT Xdir DF1:C/Xdir PURE
```

Mit diesem Befehl wird der experimentelle, nicht pure Befehl XDIR als residenter Befehl gespeichert.

Beispiel 4:

```
1> RESIDENT CD REMOVE
```

Durch diesen Befehl steht der interne Befehl CD nicht mehr zur Verfügung.

Beispiel 5:

```
1> RESIDENT CD REPLACE
```

Stellt den Befehl CD im System wieder her.

Siehe auch: PROTECT, LIST.

RUN

Befehle als Hintergrundprozesse ausführen.

Format	RUN <Befehl...> [{+ <Befehl>}]
Schablone	COMMAND/F
Pfad	Intern

Mit diesem Befehl starten Sie Hintergrundprozesse. Für einen solchen Prozeß wird kein eigenes Fenster für die Ein- und Ausgabe geöffnet, so daß auch die übergeordnete Shell freibleibt.

Mit dem Befehl RUN werden der <Befehl> und die zugehörigen Argumente ausgeführt. Sie können mit einem RUN mehrere Befehle aufrufen, indem Sie am Zeilenende ein Pluszeichen (+) eingegeben und dann die Eingabetaste drücken. Die darauf folgende Zeile wird als Fortsetzung von RUN interpretiert, aber erst nach Abarbeitung der vorhergehenden Zeile ausgeführt.

Wenn Sie das Shell-Fenster, in dem der Prozeß gestartet wurde, vor dessen Beendigung schließen wollen, können Sie die Ausgabe von RUN mit dem Befehl **`RUN >NIL: <Befehl>`** umleiten.

Die neue Hintergrund-Shell hat denselben Suchpfad und dieselbe Stack-Größe wie die Shell, in der der Befehl RUN aufgerufen wurde.

Über den Befehl RUN können Sie auch resident gespeicherte Befehle aufrufen. Um die Geschwindigkeit zu erhöhen, wird die Liste residenter Befehle vor dem Befehlspfad durchsucht. Auch für eine mit RUN NEWSHELL gestartete Shell wird die Standardstartdatei "S:Shell-startup" verwendet.

Beispiel 1:

```
1> RUN COPY Text TO PRT:+
DELETE Text +
ECHO "Druck beendet"
```

Mit diesem Befehl wird die Datei "Text" an den Drucker gesendet, ausgedruckt und gelöscht. Die angegebene Meldung wird angezeigt. Mit Pluszeichen werden Befehlszeilen verknüpft, so daß jeweils nach Abschluß eines Befehls der nächste gestartet wird.

Beispiel 2:

```
1> RUN EXECUTE Comseq
```

Durch diesen Befehl werden alle Befehle in der Befehlsdatei "Comseq" im Hintergrund abgearbeitet.

Weitere Beispiele zur Verwendung des Befehls RUN finden Sie in Kapitel 8.

SEARCH

Angegebene Zeichenkette in den Dateien der angegebenen Verzeichnisse suchen.

Format SEARCH [FROM] {<Name | Namensmuster>} [SEARCH] <Zeichenkette | Namensmuster> [ALL] [NONUM] [QUIET] [QUICK] [FILE] [PATTERN]

Schablone FROM/M,SEARCH/A,ALL/S,NONUM/S, QUIET/S,QUICK/S,FILE/S,PATTERN/S

Pfad C:

Mit diesem Befehl werden die Dateien in dem bei FROM angegebenen Verzeichnis nach der Suchzeichenkette durchsucht. (Die Schlüsselwörter FROM und SEARCH müssen nicht angegeben werden.) Wenn der Schalter ALL verwendet wird, wird auch in allen Unterverzeichnissen des bei FROM angegebenen Verzeichnisses gesucht. Die jeweils gerade durchsuchte Datei und die Zeilen, die die Suchzeichenkette enthalten, werden mit Zeilennummer angezeigt. Wenn die Suchzeichenkette Leerzeichen enthält, muß sie zwischen

Anführungszeichen gesetzt werden. Es wird nicht zwischen Groß- und Kleinschreibung unterschieden.

Folgende Optionen sind verfügbar:

NONUM	Die Zeilennummern werden nicht angezeigt.
QUIET	Der Name der gerade durchsuchten Datei wird nicht angezeigt.
QUICK	Ein kompakteres Ausgabeformat wird verwendet.
FILE	Es wird nach einer Datei mit dem angegebenen Namen gesucht und nicht nach einer Zeichenkette in einer Datei.
PATTERN	In der Suchzeichenkette können Namensmuster verwendet werden.

Wenn die Suchzeichenkette gefunden wurde, wird das Bedingungskennzeichen auf 0 gesetzt, andernfalls auf 5 (WARN). Mit Ctrl-D können Sie die Suche in der aktuellen Datei abbrechen und in der nächsten fortsetzen. Mit Ctrl-C wird die Suche vollständig abgebrochen.

Beispiele

```
1> SEARCH DEVS:DOSDrivers globvec
   DOSDrivers (dir)
      PIPE..
   6  GlobVec      =-1
      PIPE.info

1> SEARCH Utilities #?.info FILE
Workbench:Utilities/Clock.info
Workbench:Utilities/MultiView.info
```

SET

Lokale Variable definieren.

Format	SET [<Name>] [<Zeichenkette...>]
Schablone	NAME,STRING/F
Pfad	Intern

SET ohne Angabe von Argumenten liefert eine Liste der aktuellen lokalen Variablen.

Mit dem Befehl SET zusammen mit den Argumenten <Name> und <Zeichenkette> wird eine neue Umgebungsvariable definiert. Das erste Wort hinter SET wird als <Name>, der Rest der Befehlszeile als <Zeichenkette> verarbeitet. Anführungszeichen sind nicht erforderlich.

Eine mit SET definierte Umgebungsvariable kann nur in der Shell verwendet werden, in der sie definiert wurde, sowie in Shells, die mit dem Befehl NEWSHELL in der Ausgangs-Shell gestartet wurden. Wenn Sie eine Shell über das Shell-Piktogramm oder über den Menüpunkt "Befehl ausführen" öffnen, sind mit SET vorher definierte Variablen nicht verfügbar.

Sie können Umgebungsvariablen in einer Befehlsdatei oder in einer Befehlszeile aufrufen, indem Sie vor dem Variablennamen ein Dollarzeichen ($) eingeben.

Mit dem Befehl UNSET wird die Definition einer lokalen Variablen rückgängig gemacht.

Beispiel:

```
1> SET Ursprung Über Piktogramm gestarteter Prozeß
```

Mit diesem Befehl wird die Variable "Ursprung" definiert, die einen Hinweis enthält, daß eine Shell über ein Piktogramm geöffnet wurde und nicht mit dem Befehl NEWSHELL.

```
1> ECHO $Ursprung
Über Piktogramm gestarteter Prozeß
```

Siehe auch: GET, UNSET

SETCLOCK

Batteriegepufferte Echtzeituhr stellen bzw. Zeit abfragen.

Format	SETCLOCK LOAD | SAVE | RESET
Schablone	LOAD/S,SAVE/S,RESET/S
Pfad	C:

Mit dem Befehl SETCLOCK SAVE werden Datum und Uhrzeit der batteriegepufferten Echtzeituhr nach der aktuellen Systemzeit gestellt (die mit dem Time-Editor oder dem Befehl DATE gestellt wurde). SETCLOCK SAVE wird normalerweise nach dem Befehl DATE verwendet.

Mit SETCLOCK LOAD wird die aktuelle Systemzeit von der batteriegepufferten Uhr übernommen. Seit AmigaDOS Version 2 geschieht das automatisch beim Neustart.

Mit der Option RESET können Sie die Uhr zurücksetzen. Dies kann erforderlich sein, wenn die Uhr versehentlich durch einen Softwarefehler gestoppt wird oder die Befehle LOAD und SAVE nicht einwandfrei funktionieren.

Beispiel:

```
1> DATE 22-JAN-94 07:15:25
1> SETCLOCK SAVE
```

Mit diesem Befehl werden das Datum, der 22. Januar 1994, und die Uhrzeit 7:15 in der batteriegepufferten Echtzeituhr gespeichert. Die Zeit bleibt auch beim Ausschalten des Amigas erhalten. Bei einem Neustart wird die Systemuhr auf die Zeit der Echtzeituhr gestellt.

Manche Amiga-Modelle besitzen nur dann eine batteriegepufferten Echtzeituhr, wenn eine entsprechende Erweiterungskarte eingebaut ist.

Siehe auch: DATE

SETDATE

Datums- und Zeitangabe einer Datei oder eines Verzeichnisses ändern.

Format SETDATE <Datei|Namensmuster> [<Wochentag>] [<Datum>] [<Uhrzeit>] [ALL]

Schablone FILE/A,WEEKDAY,DATE,TIME,ALL/S

Pfad C:

Mit diesem Befehl wird die Datums- und Uhrzeitangabe (Datum und Uhrzeit der Erstellung bzw. der letzten Änderung) einer Datei oder eines Verzeichnisses geändert. Mit SETDATE <Datei> werden Datum und Uhrzeit auf Systemdatum und -uhrzeit eingestellt. SETDATE ALL ändert Datum und Uhrzeit aller Dateien und Unterverzeichnisse, deren Namen dem eingegebenen Namensmuster entsprechen.

SETDATE hat keinen Einfluß auf Software- und Echtzeituhr.

Die Ausgabe des Befehls DATE kann als Eingabe für den Befehl SETDATE benutzt werden.

Beispiel 1:

```
1> SETDATE Testdatei
```

Mit diesem Befehl werden Uhrzeit und Datum der Datei "Testdatei" durch das aktuelle Datum und die aktuelle Uhrzeit ersetzt.

Beispiel 2:

```
1> SETDATE Testdatei 01-04-91 13:45:32
```

Mit diesem Befehl werden Datum und Uhrzeit der Datei "Testdatei" auf den 1. April 1991, 13:45 gestellt.

Siehe auch: DATE

SETENV

Globale Variable definieren.

Format	SETENV [<Name>] [<Zeichenkette...>]
Schablone	NAME,STRING/F
Pfad	Intern

Die Angabe von SETENV ohne Argumente liefert eine Liste der aktuellen globalen Variablen.

Mit dem Befehl SETENV zusammen mit den Argumenten <Name> und <Zeichenkette> wird eine neue globale Umgebungsvariable definiert. Das erste Wort hinter SETENV wird als <Name>, der Rest der Befehlszeile als <Zeichenkette> verarbeitet. Anführungszeichen sind nicht erforderlich.

Globale Variablen werden im Verzeichnis "ENV:" als einfache Klartextdateien gespeichert und können von allen Prozessen verwendet werden. Wenn aber eine lokale Variable (definiert mit SET) und eine globale Variable denselben Namen haben, wird die lokale benutzt.

Sie können Umgebungsvariablen in einer Befehlsdatei oder in einer Befehlszeile aufrufen, indem Sie vor dem Variablennamen ein Dollarzeichen ($) eingeben.

Mit dem Befehl UNSETENV wird die Definition einer globalen Variablen gelöscht.

Beispiel 1:

```
1> SETENV Editor Extras:Tools/MEmacs
```

Mit diesem Befehl wird die Umgebungsvariable "Editor" definiert, die vom Hilfsprogramm MORE verwendet werden kann. Damit wird als Editor "MEmacs" angegeben, der sich in der Schublade "Tools" auf der Disk "Extras" befindet. Die Variable "Editor" steht in allen Shells zur Verfügung.

Beispiel 2:

```
1> SETENV Editor C:ED
```

Dieser Befehl entspricht dem obigen. Als Editor wird hier aber "ED" angegeben.

```
1> ECHO $Editor
C:ED
```

Siehe auch: GETENV, UNSETENV

SETFONT

Zeichensatz für die aktuelle Shell ändern.

Format	SETFONT <Zeichensatz> <Größe> [SCALE] [PROP] [ITALIC] [BOLD] [UNDERLINE]
Schablone	NAME/A,SIZE/N/A,SCALE/S,PROP/S, ITALIC/S,BOLD/S,UNDERLINE/S
Pfad	C:

Mit diesem Befehl können Sie den Zeichensatz für die aktuelle Shell ändern, d. h. die Standardtexteinstellungen im Font-Editor werden außer Kraft gesetzt. Der Befehl SETFONT wirkt sich nur auf das Fenster aus, in dem er aufgerufen wurde.

Bei SETFONT müssen sowohl der Name des Zeichensatzes als auch die gewünschte Schriftgröße angegeben werden. Folgende weitere Optionen stehen zur Verfügung:

SCALE	Die Skalierung von Bitmap-Zeichensätzen wird aktiviert.
PROP	Proportionale Zeichensätze werden zugelassen.
ITALIC	Die Schrift wird kursiv dargestellt.
BOLD	Die Schrift erscheint in Fettdruck.
UNDERLINE	Die Zeichen der Schrift werden unterstrichen.

Wenn SETFONT aufgerufen wird, wird der Inhalt des Shell-Fensters gelöscht, und die Eingabeaufforderung erscheint oben im Fenster in der neuen Schriftart. Von der Verwendung proportionaler Zeichensätze in einem Shell-Fenster wird abgeraten, da durch den

variablen Zeichenabstand eine spaltenweise Anzeige von Daten verhindert und das Edieren der Befehlszeile erschwert wird.

Beispiel:

```
1> SETFONT topaz 11 BOLD UNDERLINE
```

Das Shell-Fenster wird gelöscht. Die neue Eingabeaufforderung erscheint in der Schriftart Topaz, 11 Punkt hoch, in Fettdruck und unterstrichen.

SETKEYBOARD

Tastaturbelegung für die Shell ändern.

Format SETKEYBOARD <Tastaturbelegung>

Schablone KEYMAP/A

Pfad C:

Mit diesem Befehl wird die Tastaturbelegung angegeben. Folgende Tastaturtreiberdateien sind verfügbar:

Tastatur-belegung	Tastatur
cdn	Kanada (französisch)
ch1	Schweiz (französisch)
ch2	Schweiz (Deutsch)
d	Deutsch
dk	Dänisch
e	Spanisch
f	Französisch
gb	Großbritannien (Englisch)
i	Italienisch
n	Norwegisch
po	Portugiesisch
s	Schwedisch

Tastatur-belegung (Forts.)	Tastatur (Forts.)
usa	USA (Englisch)
usa2	Dvorak

Wenn Sie permanent mit einer bestimmten Tastaturbelegung arbeiten wollen, speichern Sie Ihre Auswahl über die "Preferences" (Voreinstellungen) im Input-Editor.

Beispiel:

Wenn Sie die deutsche Tastaturbelegung aufrufen wollen, geben Sie folgenden Befehl ein:

```
1> SETKEYBOARD d
```

Die Datei für die Tastaturbelegung muß sich im Verzeichnis KEYMAPS: befinden.

SKIP

Ausführung von Befehlsdateien an einer Sprungmarke fortsetzen.

Format	SKIP [<Sprungmarke>] [BACK]
Schablone	LABEL,BACK/S
Pfad	Intern

Mit diesem Befehl wird die Ausführung in einer Befehlsdatei an der <Sprungmarke> fortgesetzt. Diese wird durch die Anweisung LAB definiert. Wenn keine <Sprungmarke> angegeben ist, wird die Verarbeitung bei der nächsten LAB-Anweisung fortgesetzt.

Mit SKIP wird die Sprungmarke normalerweise ab der aktuellen Zeile vorwärts gesucht. Bei Verwendung der Option BACK beginnt die Suche aber am Dateianfang. Dadurch kann man in einer Befehlsdatei rückwärts springen, um eine Schleife zu programmieren.

Mit SKIP können Sie höchstens bis zur nächsten EXECUTE-Anweisung zurückgehen. Wenn die Befehlsdatei keine EXECUTE-Anweisung enthält, können Sie bis an den Dateianfang zurückspringen.

Wenn die angegebene Sprungmarke nicht gefunden wird, wird der Befehl abgebrochen, und die Meldung **`Label <label> not found by Skip`** (Sprungmarke nicht gefunden) wird angezeigt.

Beispiel:

Angenommen, die Befehlsdatei "Testdatei" hat folgenden Inhalt:

```
.KEY name
IF exists <Name>
   SKIP Meldung
ELSE
   ECHO "<Name> befindet sich nicht in diesem
   Verzeichnis."
   QUIT
ENDIF
LAB Meldung
ECHO "Die Datei <Name> ist vorhanden."
```

Rufen Sie die Befehlsdatei wie folgt auf:

```
1> EXECUTE Testdatei Dokument
```

Wenn sich die Datei "Dokument" im aktuellen Verzeichnis befindet, wird durch den Befehl SKIP die Ausführung bei dem Befehl LAB fortgesetzt. Folgende Meldung wird im Shell-Fenster angezeigt:

```
Die Datei Dokumentdatei ist vorhanden.
```

Wenn sich die Datei "Dokument" nicht im aktuellen Verzeichnis befindet, wird die Ausführung Ausführung des Skripts in der Zeile nach der Anweisung ELSE fortgesetzt, und die folgende Meldung wird im Shell-Fenster angezeigt:

```
Dokument befindet sich nicht in diesem Verzeichnis.
```

Siehe auch: EXECUTE, LAB. Weitere Beispiele zum Befehl SKIP finden Sie in Kapitel 8.

SORT

Zeilen einer Datei alphabetisch sortieren.

Format	SORT [FROM] <Datei | Namensmuster> [TO] <Dateiname> [*COLSTART* <n>] [CASE] [NUMERIC]
Schablone	FROM/A,TO/A,COLSTART/K,CASE/S,NUMERIC/S
Pfad	C:

Mit diesem Befehl werden die Zeilen der bei FROM angegebenen Datei alphabetisch sortiert. Das Ergebnis wird in der bei TO angegebenen Datei gespeichert. Es wird vorausgesetzt, daß es sich um eine normale Textdatei handelt, deren Zeilen durch Zeilenvorschubzeichen voneinander getrennt sind. Normalerweise wird zwischen Groß- und Kleinschreibung nicht unterschieden. Bei Verwendung des Schalters CASE werden aber Groß- vor Kleinbuchstaben einsortiert.

Mit dem Schlüsselwort COLSTART können Sie die Spalte angeben, in der der Vergleich beginnen soll. Von dieser Stelle an werden die Zeilen verglichen. Wenn die verglichenen Zeilen bis zum Zeilenende übereinstimmen, wird der Vergleich am Zeilenanfang fortgesetzt.

Mit der Option NUMERIC werden Zahlen sortiert. Der Vergleich beginnt in der ersten Spalte und wird nach rechts bis zum ersten Zeichen, das keine Ziffer ist, fortgesetzt. Zeilen, die nicht mit einer Ziffer beginnen, werden als 0 einsortiert. Die Zeilen werden in numerischer Reihenfolge ausgegeben. Wenn bei NUMERIC der Schalter CASE angegeben wird, wird er ignoriert.

Beispiel:

```
1> SORT DF0:Glossar TO DF0:Glossar.sort
```

Mit diesem Befehl werden die Zeilen in der Datei "Glossar" alphabetisch sortiert. Die Ausgabe erfolgt in die Datei "Glossar.sort". Bei dem Sortiervorgang wird nicht zwischen Groß- und Kleinschreibung unterschieden.

Weitere Beispiele zum Befehl SORT finden Sie in Kapitel 8.

STACK

Stack-Größe innerhalb der aktuellen Shell anzeigen oder festlegen.

Format	STACK [[SIZE] <Stack-Größe>]
Schablone	SIZE/N
Pfad	Intern

Eine Shell belegt eine bestimmte Menge Stack-Speicher, ein spezieller, der Shell zugeordneter Bereich des Hauptspeichers. Jede Shell besitzt eine spezifische Stack-Größe. Wenn ein Programm einen Systemfehler verursacht, können Sie es u. U. durch Vergrößern des Stacks der betreffenden Shell doch noch zum Funktionieren bringen. Befehle, die Funktionen ausführen, die in mehreren Ebenen verschachtelt sind, können einen größeren Stack-Umfang erforderlich machen.

Die Größe des Stack-Speichers liegt normalerweise zwischen 4.096 und 40.000 Byte. Wenn die Stack-Größe nicht ausreicht, kann ein Systemfehler auftreten. Wenn der Stack-Speicher jedoch zu groß ist, wird dadurch für andere Systemfunktionen benötigter Speicherplatz blockiert.

Hinweis: Wenn der Stack-Speicher nicht groß genug ist, erhalten Sie unter Umständen eine Softwarefehlermeldung. Erhöhen Sie die Stack-Größe für die Shell, die den Fehler verursacht hat.

Wenn Sie den Befehl STACK ohne Argumente eingeben, wird die aktuelle Stack-Größe angezeigt.

STATUS

Informationen über Shell-Prozesse anzeigen.

Format STATUS [<Prozeß>] [FULL] [TCB] [CLI | ALL] [*COM* | *COMMAND* <Befehl>]

Schablone PROCESS/N,FULL/S,TCB/S,CLI=ALL/S, COM=COMMAND/K

Pfad C:

Wenn der Befehl ohne Argumente eingegeben wird, werden die Nummern der aktuellen Shell-Prozesse und gegebenenfalls die darunter laufenden Programme und Befehle angezeigt. Mit dem Argument <Prozeß> wird die Prozeßnummer angegeben, wodurch nur Informationen zu diesem einen Prozeß angezeigt werden.

Wenn Sie Informationen über die Stack-Größe, die Werte der globalen Vektoren, die Priorität und die aktuellen Befehle für jeden Prozeß abrufen wollen, benutzen Sie das Schlüsselwort FULL. Die Funktion des Schlüsselworts TCB ist ähnlich, führt aber keine Informationen über die Befehle auf. Mit dem Schlüsselwort CLI=ALL erhalten Sie dagegen ausschließlich Informationen über die Befehle.

Mit der Option COMMAND wird nach einem bestimmten Befehl gesucht. Die Shell-Liste wird dann nach dem angegebenen <Befehl> durchsucht. Wenn er gefunden wird, wird die Shell-Nummer ausgegeben und das Bedingungskennzeichen auf 0 gesetzt. Andernfalls wird es auf 5 (WARN) gesetzt.

Beispiel 1:

```
1> STATUS 1
Process 1: Loaded as command: status
```

Beispiel 2:

```
1> STATUS 1 FULL
Process 1: stk 4000, gv 150, pri 0 Loaded as
command: status
```

Beispiel 3:

```
1> STATUS >RAM:Xyz COMMAND=COPY
1> BREAK <RAM:Xyz >NIL: ?
```

Der Prozeß, in dem der Befehl COPY ausgeführt wird, wird mit BREAK unterbrochen.

TYPE

Inhalt einer Datei anzeigen.

Format	TYPE {<Datei|Namensmuster>} [*TO* <Name>] [*OPT* H|N] [HEX|NUMBER]
Schablone	FROM/A/M,TO/K,OPT/K,HEX/S,NUMBER/S
Pfad	C:

Mit diesem Befehl wird der Inhalt der angegebenen Datei im aktuellen Fenster angezeigt, wenn kein Ziel angegeben wird, bzw. in der angegebenen Datei gespeichert. Mit dem Schlüsselwort TO werden die Daten in eine angegebene Datei geschrieben. Wenn mehr als ein Dateiname angegeben wird, wird der Inhalt der Dateien nacheinander angezeigt.

Die Optionen OPT H und OPT N stehen auch in Form der Schlüsselwörter HEX bzw. NUMBER zur Verfügung. Diese beiden Optionen schließen sich gegenseitig aus. Mit der Option HEX wird der Inhalt der Datei als Hexadezimalzahlen in Spalten ausgegeben. Eine weitere Spalte zeigt jeweils das zugehörige ASCII-Zeichen. Dies ist vor allem bei der Analyse von Binärdateien von Vorteil. Mit der Option NUMBER werden die Zeilen bei der Ausgabe numeriert.

Wenn Sie die Ausgabe unterbrechen wollen, drücken Sie die Leertaste. Mit der Rücktaste, der Eingabetaste oder Ctrl-X können Sie die Ausgabe fortsetzen. Wenn Sie die Ausgabe abbrechen wollen, drücken Sie Ctrl-C (`***Break` (ABBRUCH) wird angezeigt).

Beispiel:

```
1> TYPE S:Startup-sequence
```

Der Inhalt der Datei "Startup-sequence" im Verzeichnis S: wird angezeigt.

Weitere Beispiele zum Befehl TYPE finden Sie in Kapitel 8.

UNALIAS

Alias-Namen löschen.

Format	UNALIAS [<Name>]
Schablone	NAME
Pfad	Intern

Mit diesem Befehl wird der angegebene Alias aus der Aliasliste gelöscht. Wenn UNALIAS ohne Argumente angegeben wird, werden die aktuellen Aliase angezeigt.

Siehe auch: ALIAS

UNSET

Lokale Variable löschen.

Format	UNSET [<Name>]
Schablone	NAME
Pfad	Intern

Mit diesem Befehl wird die angegebene lokale Variable aus der Variablenliste des aktuellen Prozesses gelöscht. Wenn UNSET ohne Argumente angegeben wird, werden die aktuellen Variablen angezeigt.

Siehe auch: SET

UNSETENV

Global Variable löschen.

Format	UNSETENV [<Name>]
Schablone	NAME
Pfad	Intern

Mit diesem Befehl wird die angegebene globale Variable aus der aktuellen Variablenliste gelöscht. Wenn UNSETENV ohne Argumente angegeben wird, werden die aktuellen Variablen angezeigt.

Siehe auch: SETENV

VERSION

Versions- und Revisionsnummern von Software abfragen.

Format	VERSION [<Bibliothek | Gerät | Datei>] [<Versionsnr.>] [<Revisionsnr.>] [<Einheitennr.>] [FILE] [INTERNAL] [RES] [FULL]
Schablone	NAME,VERSION/N,REVISION/N,UNIT/N, FILE/S,INTERNAL/S,RES/S,FULL/S
Pfad	C:

Mit diesem Befehl werden die Versions- und die Revisionsnummer einer Bibliothek, eines Gerätetreibers, eines Befehls oder einer Workbench-Disk ermittelt. Auch eine Überprüfung auf eine bestimmte Versions- bzw. Revisionsnummer ist möglich, wobei das Bedingungskennzeichen entsprechend gesetzt wird, wenn die Nummer größer ist.

Wenn der Befehl VERSION ohne das Argument <Bibliothek | Gerät | Datei> verwendet wird, wird die Nummer der Kickstart- und der Workbench-Version ausgegeben, und diese beiden Umgebungsvariablen werden definiert. Wenn ein Name angegeben wird, wird die Bibliothek, der Gerätetreiber, das Laufwerk bzw. die Datei geöffnet, und die Versionsnummer wird gelesen. Die Versionsnummer des Dateisystems können Sie abrufen, indem Sie einen Laufwerksnamen eingeben (z. B. DF0: oder DH0:).

Wenn eine <Versionsnr.> (und eventuell eine <Revisionsnr.>) angegeben wird, wird das Bedingungskennzeichen auf 0 gesetzt, wenn die Versionsnummer (bzw. Revisionsnummer) des Kickstarts, der Bibliothek oder des Gerätetreibers größer oder gleich dem angegeben Wert ist. Andernfalls wird das Bedingungskennzeichen auf 5 (WARN) gesetzt. (Wenn keine Revisionsnummer angegeben wird, wird sie auch nicht verglichen.)

Die Option <Einheitennr.> ist überflüssig und wird nur aus Gründen der Kompatibilität zu älteren Programmen beibehalten.

Mit der Option FILE zwingen Sie den Befehl VERSION, die zur Zeit geladenen Bibliotheken oder Gerätetreiber zu ignorieren. Das gibt Ihnen die Möglichkeit, die Versionsnummer einer Datei .library oder .device auf einer Disk abzurufen, obwohl bereits eine Bibliothek bzw. eine Einheit dieses Namens im Hauptspeicher geladen oder in LIBS: verfügbar ist. Die Option RES ruft die Version der residenten Befehle ab. In Shells integrierte Befehle haben dieselbe Versionszeichenkette wie die Shell selbst. INTERNAL ist ebenfalls überflüssig und wird nur aus Gründen der Kompatibilität beibehalten. Mit der Option FULL werden alle Angaben zur Version (einschließlich Datum) ausgegeben.

Beispiel:

```
1> VERSION
Kickstart 39.92 Workbench 39.1

1> VERSION Alpha:Libs/xyz.library FILE FULL
xyz.library 1.13 (18.02.93)
```

WAIT

Angegebenen Zeitraum warten.

Format	WAIT [<n>] [SEC|SECS|MIN|MINS] [*UNTIL* <Uhrzeit>]
Schablone	/N,SEC=SECS/S,MIN=MINS/S,UNTIL/K
Pfad	C:

Pointer

Mit diesem Befehl wird das Aussehen des Mauspfeils geändert.

Format POINTER [FROM <Dateiname>] [EDIT | USE | SAVE] [*CLIPUNIT* <Zwischenspeichereinheit>] [NOREMAP]

Schablone FROM,EDIT/S,USE/S,SAVE/S,CLIPUNIT/K/N, NOREMAP/S

Pfad Extras:Prefs

Beispiel:

```
1> POINTER CLIPUNIT 1
```

Der Mauspfeil-Voreinsteller (Pointer) wird geöffnet und die Zwischenspeichereinheit auf 1 eingestellt. Dies ist empfehlenswert, wenn die standardmäßige Zwischenspeichereinheit (0) bereits benutzt wird.

Printer

Mit diesem Befehl werden ein Drucker und Druckoptionen angegeben.

Format PRINTER [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>] [UNIT]

Schablone FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K, UNIT/S

Pfad Extras:Prefs

Beispiel:

```
1> PRINTER Prefs/Presets/Printer.post UNIT
```

Der Drucker-Voreinsteller (Printer) wird geöffnet, und die Einstellungen der Voreinstellungsdatei "Printer.post" werden geladen. Im Voreinsteller wird ein Textfeld angezeigt, in dem die

Overscan

Mit diesem Befehl wird die Größe der Anzeigebereiche für Text und Grafiken geändert.

Format	OVERSCAN [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]
Schablone	FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K
Pfad	Extras:Prefs

Beispiel:

```
1> OVERSCAN PUBSCREEN MeinSchirm
```

Der Randbereichs-Voreinsteller (Overscan) wird auf dem Public-Schirm "MeinSchirm" geöffnet.

Palette

Mit diesem Befehl werden die Farben des Workbench-Bildschirms geändert.

Format	PALETTE [FROM <Dateiname>] [EDIT | USE | SAVE]
Schablone	FROM,EDIT/S,USE/S,SAVE/S
Pfad	Extras:Prefs

Input

Mit diesem Befehl werden die Geschwindigkeiten für Maus und Tastatur eingestellt sowie ein länderspezifischer Tastaturtreiber ausgewählt.

Format INPUT [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]

Schablone FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K

Pfad Extras:Prefs

Beispiel:

```
1> INPUT Prefs/Presets/Input.fast USE
```

Mit diesem Befehl werden die Einstellungen, die in der Voreinstellungsdatei "Input.fast" gespeichert wurden, ohne Öffnen des Voreinstellers geladen und aktiviert. Bei einem Neustart des Systems werden wieder die zuvor gespeicherten Standardeinstellungen verwendet.

Locale

Mit diesem Befehl können Sie die Sprachen auswählen, die auf Ihrem System verfügbar sein sollen.

Format LOCALE [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]

Schablone FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K

Pfad Extras:Prefs

Beispiel:

```
1> LOCALE Prefs/Presets/Locale.UK SAVE
```

Die Einstellungen der Voreinstellungsdatei "Locale.UK" werden geladen und ohne Öffnen des Voreinstellers als Standardeinstellungen gespeichert. Diese neuen Einstellungen bleiben bei einem späteren Neustart des Computers erhalten.

Font

Mit diesem Befehl werden die vom System zu benutzenden Zeichensätze angegeben.

Format	FONT [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]
Schablone	FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K
Pfad	Extras:Prefs

Beispiel:

```
1> FONT
```

Der Zeichensatz-Voreinsteller (Font) wird geöffnet. Dieser Befehl entspricht dem doppelten Anklicken des Piktogramms "Font".

IControl

Mit diesem Befehl werden einige vom Betriebssystem benutzten Parameter angegeben.

Format	ICONTROL [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]
Schablone	FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K
Pfad	Extras:Prefs

Beispiel:

```
1> ICONTROL Prefs/Presets/IControl.pre
```

Der Voreinsteller IControl wird geöffnet. Gleichzeitig werden die Einstellungen aus der Voreinstellungsdatei "IControl.pre" zum Edieren geladen.

7.1 Voreinsteller-Editoren

Über die in diesem Kapitel aufgeführten Befehle werden die Voreinsteller-Editoren der Workbench aufgerufen.

In den Befehlsformaten für viele Voreinsteller-Editoren werden immer wieder die gleichen Argumente und Schalter genannt. In allen Fällen haben sie dieselbe Bedeutung. Diese können Sie der folgenden Tabelle entnehmen:

Argument	Bedeutung
[FROM <Dateiname>]	Gibt an, daß eine bereits konfigurierte Voreinstellungsdatei geöffnet werden soll. Diese Datei muß zuvor über den Menüpunkt "speichern als" des jeweiligen Voreinstellers gespeichert worden sein. Diese Dateien haben normalerweise die Namenserweiterung ".pre" und befinden sich in der Schublade "Presets" (Voreinstellungen).
[EDIT]	Der Voreinsteller wird geöffnet. Dies ist die Vorgabe, wenn Sie nur den Namen des Voreinstellers eingeben.
[USE]	Die Einstellungen in der mit FROM angegebenen Datei werden ohne Öffnen des Voreinstellers verwendet.
[SAVE]	Die Einstellungen in der mit FROM angegebenen Datei werden ohne Öffnen des Voreinstellers als Vorgaben gespeichert.
[PUBSCREEN <Public-Schirm>]	Ermöglicht dem Voreinsteller, das zugehörige Fenster auf einem Public-Schirm zu öffnen.
[UNIT]	Ein zusätzliches Textfeld zur Einstellung der Standardeinheitennummer wird im Voreinsteller-Fenster angezeigt.
[CLIPUNIT <Zwischen-speichereinheit>]	Legt fest, welche Zwischenspeichereinheit beim Kopieren und Einfügen verwendet wird.
[NOREMAP]	Die Farbzuordnung wird inaktiviert, so daß das System Bilddateien in den Farben anzeigt, mit denen sie gespeichert wurden.

Befehl (Forts.)	Funktion (Forts.)	Seite (Forts.)
MEmacs	Text bildschirmorientiert edieren.	7-28
More	Inhalt einer ASCII-Datei anzeigen.	7-28
MouseBlanker	Mauspfeil ausblenden, wenn Daten über die Tastatur eingegeben werden.	7-17
MultiView	Grafik-, Text-, AmigaGuide-Hilfs-, Sound- und Animationsdateien anzeigen.	7-30
NoCapsLock	Taste "Caps Lock" (Nur Großbuchstaben) inaktivieren.	7-17
NoFastMem	Amiga zur ausschließlichen Verwendung des Chip-RAM zwingen.	7-33
Overscan	Größen der Anzeigebereiche für Text und Grafiken ändern.	7-7
Palette	Farben des Workbench-Bildschirms ändern.	7-7
Pointer	Aussehen des Mauspfeils ändern.	7-8
PrepCard	PCMCIA-Speicherkarten zur Verwendung als Disk oder RAM vorbereiten.	7-34
Printer	Drucker und grundlegende Druckoptionen angeben.	7-8
PrinterGfx	Optionen für Grafikdrucker angeben.	7-9
PrinterPS	Optionen für PostScript-Drucker angeben.	7-9
ScreenMode	Bildschirmmodus auswählen.	7-10
Serial	Spezifikationen für die Kommunikation über die serielle Schnittstelle definieren.	7-10
Sound	Ton für akustische Signale des Bildschirms einstellen.	7-11
Time	Systemuhr stellen.	7-11
WBPattern	Hintergrundmuster für die Workbench und für Fenster erstellen.	7-12

Befehl (Forts.)	Funktion (Forts.)	Seite (Forts.)
Calculator	Taschenrechner auf dem Bildschirm anzeigen.	7-18
ClickToFront	Fenster durch Anklicken mit der Maus in den Vordergrund holen.	7-15
Clock	Uhr auf dem Bildschirm anzeigen.	7-19
CMD	Druckerausgabe in Datei umleiten.	7-21
CrossDOS	Textfilter- und Konvertierungsoptionen für CrossDOS-Geräte einstellen.	7-15
DiskCopy	Inhalt einer Disk auf eine andere kopieren.	7-22
Exchange	Commodity-Exchange-Programme steuern und überwachen.	7-16
FixFonts	.font-Dateien im Verzeichnis FONTS: aktualisieren.	7-23
FKey	Funktionstasten mit Befehlen belegen.	7-16
Font	Die vom System zu benutzenden Zeichensätze angeben.	7-5
Format	Disk für die Verwendung auf dem Amiga formatieren.	7-23
GraphicDump	Im Vordergrund befindlichen Bildschirm drucken.	7-25
IconEdit	Aussehen und Typ von Piktogrammen edieren.	7-26
IControl	Von der Workbench benutzte Parameter angeben.	7-5
InitPrinter	Drucker auf Druckoptionen initialisieren, die in den Voreinsteller-Editoren angegeben wurden.	7-27
Input	Geschwindigkeiten für Maus und Tastatur angeben sowie länderspezifische Tastaturtreiber auswählen.	7-6
Intellifont	Intellifont-Umriß-Zeichensätze verwalten.	7-27
KeyShow	Aktuelle Tastaturbelegung anzeigen.	7-27
Locale	Sprache zur Kommunikation mit dem System auswählen.	7-6

Kapitel 7

Referenz für Workbench-bezogene Befehle

Die in diesem Abschnitt beschriebenen Befehle sind die Befehlszeilenentsprechungen für Programme, die über die Workbench aufgerufen werden. Diese Befehle lassen sich in die folgenden Funktionskategorien unterteilen:

- Voreinsteller-Editoren
- Commodity-Exchange-Programme
- Weitere Workbench-bezogene (Hilfs-)Programme

Diese Befehlsunterteilung dient nur zu Dokumentationszwecken.

Hinweis Eine vollständige Beschreibung der Workbench-Editoren, -Hilfsprogramme und -Programme können Sie dem *Workbench-Benutzerhandbuch* entnehmen.

Die folgende Tabelle enthält kurze, alphabetisch sortierte Referenzangaben zu den in diesem Kapitel beschriebenen Befehlen, deren Funktion und der Seite, auf der der jeweilige Befehl erläutert wird:

Befehl	Funktion	Seite
AutoPoint	Fenster, auf dem sich der Mauspfeil befindet, automatisch auswählen.	7-13
Blanker	Bildschirmanzeige ausblenden, wenn über einen bestimmten Zeitraum keine Eingaben empfangen werden.	7-14

SETPATCH

Zum ROM-Teil der Systemsoftware werden Änderungen aktiviert.

Format	SETPATCH [QUIET] [NOCACHE] [REVERSE]
Schablone	QUIET/S,NOCACHE/S,REVERSE/S
Pfad	C:

Mit diesem Befehl werden am Betriebssystem vorübergehende Änderungen vorgenommen. Wenn vorhanden muß er am Anfang der Datei "Startup-sequence" stehen. Der Befehl SETPATCH wird im Zuge der Weiterentwicklung von AmigaDOS erforderlichenfalls angepaßt.

Mit der Option QUIET wird die Bildschirmausgabe unterdrückt.

Mit der Option NOCACHE wird verhindert, daß auf bestimmten 68030er und 68040er Systemen der Daten-Cache aktiviert wird.

Mit der Option REVERSE werden die Änderungen in umgekehrter Reihenfolge gespeichert. Diese Option wird vor allem von CDTV-Entwicklern verwendet.

die neue Einheitennummer ein. Beim nächsten Kopieren und Einfügen wird diese Zwischenspeichereinheit benutzt.

Die Verwendung der Option OFF zusammen mit Shell, MEmacs und ED führt dazu, daß diese Befehle keinen Dialogbetrieb mit der Zwischenablage mehr führen, wenn Datenelemente ausgeschnitten und eingefügt werden. Von der Verwendung dieser Option wird abgeraten.

IPREFS

In den einzelnen Editor-Dateien gespeicherte Voreinstellungsinformationen werden an das Betriebssystem übertragen.

Format	IPREFS
Schablone	(keine)
Pfad	C:

IPREFS liest die einzelnen Dateien der Systemvoreinstellung und gibt diese Daten an das System weiter. IPREFS wird normalerweise in der Startsequenz gestartet, nachdem die Voreinstellungsdateien in das Verzeichnis ENV: kopiert wurden. Jedesmal wenn ein Benutzer die Menüpunkte "Speichern" oder "Benutzen" in einem Voreinstellungseditor auswählt, werden durch IPREFS die Informationen dann an das System weitergeleitet. Wenn nötig, setzt IPREFS die Workbench zurück, um die Änderungen zu implementieren. Wenn Shell-, Projekt- oder Programm-Fenster geöffnet sind, wird ein Dialogfenster angezeigt, durch das Sie aufgefordert werden, diese Fenster zu schließen.

BINDDRIVERS

Gerätetreiber werden der entsprechenden Hardware zugeordnet.

Format	BINDDRIVERS
Schablone	(keine)
Pfad	C:

Mit diesem Befehl werden Gerätetreiber für zusätzliche Hardware, die automatisch durch die Erweiterungsbibliothek konfiguriert wird, geladen und gestartet. Diese Gerätetreiber müssen im Verzeichnis "SYS:Expansion" abgelegt sein, und zwar immer mit einer zusätzlichen .info-Datei.

Der Befehl BINDDRIVERS muß in der Datei "Startup-sequence" angegeben sein, damit die Hardware beim Systemstart automatisch konfiguriert wird.

CONCLIP

Übertragen von Daten zwischen Konsolenfenstern und der Zwischenablage (Clipboard).

Format	CONCLIP [CLIPUNIT | UNIT <Einheitennummer>] [OFF]
Schablone	CLIPUNIT=UNIT/N,OFF/S
Pfad	C:

CONCLIP wird von der standardmäßigen Startsequenz aufgerufen. Der Befehl überwacht die Daten, die in die Zwischenablage geschrieben wurden.

Mit der Option CLIPUNIT können Sie die Einheitennummer für clipboard.device angeben. Sie können eine Einheit von 0 bis 255 angeben. Vorgegeben ist die Nummer 0. Diese Option wird vor allem von fortgeschrittenen Benutzern und von Programmierern verwendet, die verschiedene Einheiten für unterschiedliche Daten benutzen wollen (z. B. eine für Text und eine andere für Grafiken). Rufen Sie dazu nur von der Shell aus den Befehl auf, und geben Sie

6.3 Systembefehle

Systembefehle werden für den normalen Systembetrieb benötigt. Sie werden von der standardmäßigen Startup-sequence verwendet oder vom System automatisch für Anwendungen aufgerufen. Der Benutzer braucht solche Befehle in der Regel nicht selbst aufzurufen.

ADDDATATYPES

Erstellen einer Liste von Datentypen, die die "datatypes.library" interpretieren kann.

Format ADDDATATYPES [FILES] {Dateinamen} [QUIET] [REFRESH]

Schablone FILES/M,QUIET/S,REFRESH/S

Pfad C:

Die Deskriptordateien der Dateitypen werden unter "DEVS:DataTypes" gespeichert. Diese Deskriptoren ermöglichen es Programmen wie z. B. MultiView, unterschiedliche Dateitypen zu interpretieren. Dieser Befehl kann auch von Befehlsdateien zur Installation von Anwendungen aufgerufen werden, die dann der Liste eigene Datentypen hinzufügen.

Mit der Option FILES wird der Name der Dateitypdeskriptoren, mit denen die Liste ergänzt werden soll, angegeben.

Mit der Option QUIET werden Fehler- und Ausgabemeldungen unterdrückt.

Mit der Option REFRESH wird das Verzeichnis "DEVS:DataTypes" auf neue oder geänderte Datentypdeskriptoren geprüft.

kennzeichen auf 5 (WARN) gesetzt, es wird aber keine Fehlermeldung ausgegeben.

Mit der Option NORES wird in der Liste der residenten Befehle nicht gesucht. Mit RES wird dagegen nur in dieser Liste gesucht.

Mit dem Schalter ALL wird die Suche auch dann fortgesetzt, wenn das gesuchte Objekt bereits gefunden wurde. Dadurch können alle Versionen eines Befehls oder eines Programms gefunden werden. Durch diesen Schalter kann derselbe Befehl aber auch mehrmals angezeigt werden, wenn er auf verschiedene Weisen aufgerufen werden kann (z. B. über C: und das aktuelle Verzeichnis).

Beispiels:

```
1> WHICH avail
C:Avail
```

```
1> WHICH C:
Workbench:C
```

```
1> WHICH alias
INTERNAL alias
```

WHY

Fehlermeldung ausgeben, in der das Fehlschlagen des vorhergehenden Befehls erläutert wird.

Format WHY

Schablone (keine)

Pfad Intern

Normalerweise wird eine kurze Meldung angezeigt, wenn ein Befehl nicht erfolgreich ausgeführt wurde. Im allgemeinen wird dabei der Name der Datei (wenn dies die Fehlerursache war) angegeben. Es werden jedoch keine weiteren Einzelheiten aufgeführt. Wenn die Ursache der Fehlermeldung unklar ist, können Sie mit WHY u. U. eine ausführlichere Erläuterung abrufen.

Mit diesem Befehl wird eine Befehlsdatei eine bestimmte Zeit lang oder bis zu einem bestimmten Zeitpunkt unterbrochen. Wenn nichts anderes angegeben ist, beträgt die Wartezeit eine Sekunde.

Mit dem Argument <n> wird die Anzahl der Sekunden (bzw. Minuten, wenn die Option MIN verwendet wird), die die Wartezeit betragen soll, angegeben. Diese Optionen schließen sich gegenseitig aus, es können also nur Minuten oder Sekunden angegeben werden.

Mit dem Schlüsselwort UNTIL dauert die Wartezeit bis zum angegebenen Zeitpunkt, der im Format HH:MM (Stunden:Minuten) einzugeben ist.

Beispiel 1:

```
1> WAIT 10 MINS
```

Mit diesem Befehl beträgt die Wartezeit 10 Minuten.

Beispiel 2:

```
1> WAIT UNTIL 21:15
```

Mit diesem Befehl dauert die Wartezeit bis 21.15 Uhr.

WHICH

Suchpfad nach einem bestimmten Objekt durchsuchen.

Format	WHICH <Befehl> [NORES] [RES] [ALL]
Schablone	FILE/A,NORES/S,RES/S,ALL/S
Pfad	C:

Mit diesem Befehl können Sie einen bestimmten Befehl, ein Programm oder ein Verzeichnis suchen. Wenn sich das Element im Suchpfad befindet, wird der vollständige Pfad angezeigt. Residente Befehle werden mit RESIDENT und interne Befehle mit INTERNAL gekennzeichnet.

Normalerweise wird in der Liste der residenten Befehle, im aktuellen Verzeichnis, in den Suchpfaden sowie im Verzeichnis C: gesucht. Wenn die Datei nicht gefunden wird, wird das Bedingungs-

Standarddruckereinheit für die Voreinstellungen angegeben werden kann.

PrinterGfx

Mit diesem Befehl werden Optionen für Grafikdrucker angegeben.

Format PRINTERGFX [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]

Schablone FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K

Pfad Extras:Prefs

PrinterPS

Mit diesem Befehl werden die Optionen für Postscript-Drucker angegeben.

Format PRINTERPS [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]

Schablone FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K

Pfad Extras:Prefs

Dieser Voreinsteller kann nur verwendet werden, wenn Sie über einen PostScript-Drucker verfügen und im Drucker-Voreinsteller (Printer) die Option "PostScript" auswählen.

ScreenMode

Mit diesem Befehl wird ein Bildschirmmodus für den Workbench-Bildschirm ausgewählt.

Format	SCREENMODE [FROM <Dateiname>] [EDIT | USE | SAVE]
Schablone	FROM,EDIT/S,USE/S,SAVE/S
Pfad	Extras:Prefs

Beispiel:

```
1> SCREENMODE Prefs/Presets/DTPscreen USE
```

Sie werden aufgefordert, alle Fenster zu schließen, die keine Schubladen enthalten. Das System wird erneut gestartet, und die in der DTPscreen-Datei gespeicherten Einstellungen werden aktiviert. Das Voreinsteller-Fenster wird nicht geöffnet. Nach einem Neustart wird für den Bildschirmmodus (bzw. Anzeigemodus) wieder die zuletzt gespeicherte Einstellung verwendet.

Serial

Mit diesem Befehl werden die Spezifikationen für die Kommunikation über die serielle Schnittstelle definiert.

Format	SERIAL [FROM <Dateiname>] [EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>] [UNIT]
Schablone	FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K, UNIT/S
Pfad	Extras:Prefs

Beispiel:

```
1> SERIAL Prefs/Presets/Serial.9600 PUBSCREEN
MidTerm UNIT
```

Der Seriell-Voreinsteller (Serial) wird geöffnet. Dabei werden die Einstellungen aus der Datei "Serial.9600" geladen. Das Voreinsteller-Fenster wird auf dem Public-Schirm MidTerm geladen. Das Fenster enthält das Eingabefeld "Vorgabe-Einheit".

Sound

Mit diesem Befehl werden der Typ der vom Amiga ausgegebenen akustischen Signale (Sound) sowie deren Attribute festgelegt.

Format	SOUND [FROM <Dateiname>][EDIT | USE | SAVE] [*PUBSCREEN* <Public-Schirm>]
Schablone	FROM,EDIT/S USE/S SAVE/S PUBSCREEN/K
Pfad	Extras:Prefs

Time

Mit diesem Befehl wird die Systemuhr gestellt..

Format	TIME [EDIT | SAVE] [*PUBSCREEN* <Public-Schirm>]
Schablone	EDIT/S,SAVE/S,PUBSCREEN/K
Pfad	Extras:Prefs

Da beim Stellen der Systemuhr über die Shell die neuen Werte stets gespeichert werden, entfällt beim Zeit-Voreinsteller die Option "Benutzen" (USE).

WBPattern

Mit diesem Befehl werden Hintergrundmuster für die Workbench und die Workbench-Fenster erstellt.

Format	WBPATTERN [FROM <Dateiname>] [EDIT | USE | SAVE] [*CLIPUNIT* <Zwischenspeichereinheit>] [NOREMAP]
Schablone	FROM,EDIT/S,USE/S,SAVE/S,CLIPUNIT/K/N, NOREMAP/S
Pfad	Extras:Prefs

Beispiel:

```
1> WBPattern Prefs/Presets/Tapete.pattern NOREMAP
```

Der Workbench-Muster-Voreinsteller (WBPattern) wird geöffnet. Dabei werden die Einstellungen aus der Datei "Tapete.pattern" geladen. Durch die Option NOREMAP wird verhindert, daß die Farben in geladenen Bildern und Mustern neu zugeordnet werden.

7.2 Commodity-Exchange-Programme

Mit den folgenden Befehlen werden die Commodity-Exchange-Programme der Workbench aufgerufen. Die Programme befinden sich im Verzeichnis "Tools/Commodities". Weitere Informationen finden Sie im Workbench-Handbuch.

Die folgenden Argumente in Befehlszeilen für Commodity-Exchange-Programme haben in allen Befehlen, in denen sie verwendet werden, dieselbe Bedeutung. Sie entsprechen der Eingabe der jeweiligen Merkmale im Fenster "Information" für ein Piktogramm.

Argument	Bedeutung
[CX_PRIORITY <Priorität>]	Gibt die Priorität des Commodity-Exchange-Programms in bezug auf alle anderen Commodity-Exchange-Programme an. Der Standardwert ist 0. Wenn Sie für <Priorität> einen Wert größer als 0 angeben, erhält das Programm Priorität gegenüber anderen Commmodity-Exchange-Programmen in bezug auf Tastenbefehle und andere Commodity-bezogene Dinge. Die Einstellung hat keine Auswirkung auf Task-Prioritäten.
[CX_POPKEY <Taste>]	Mit diesem Argument können Sie den Tastenbefehl angeben (falls vorhanden), mit dem ein Programmfenster geöffnet wird. Bei Angabe von mehreren Tasten müssen Sie die Tasten in doppelte Anführungszeichen setzen (Details siehe im Workbench-Handbuch, z. B. CX_POPKEY="Shift F1").
[CX_POPUP= <yes/no>]	Gibt an, ob das Programmfenster bei Eingabe des Befehls geöffnet werden soll.

AutoPoint

Bei Verwendung dieses Befehls werden Fenster, auf denen sich der Mauszeiger befindet, automatisch geöffnet.

Format AUTOPOINT [*CX_PRIORITY* <Priorität>]

Schablone CX_PRIORITY/N/K

Pfad Extras:Tools/Commodities

Wenn Sie AUTOPOINT von einer Shell aus aufgerufen haben, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

Blanker

BLANKER ist ein Commodity-Exchange-Programm, mit dem der Bildschirm ausgeblendet bzw. eine Animation angezeigt wird, wenn innerhalb des angegebenen Zeitraums keine Eingabe erfolgt. Dies verhindert das Einbrennen des Monitors.

Format BLANKER [*CX_PRIORITY* <Priorität>] [*CX_POPKEY* <Taste>] [*CX_POPUP*=<yes | no>] [*SECONDS* <Wartezeit>] [*CYCLECOLORS* <yes | no>] [*ANIMATION* <yes | no>]

Schablone CX_PRIORITY/N/K,CX_POPKEY/K,CX_POPUP/K,SECONDS/N/K,CYCLECOLORS/K,ANIMATION/K

Pfad Extras:Tools/Commodities

Die Argumente für BLANKER entsprechen den Merkmalen im Fenster "Information" des Piktogramms "Blanker".

Wenn Sie das Programm von einer Shell aus aufgerufen haben, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

Beispiel 1:

```
1> BLANKER SECONDS 45
```

Das BLANKER-Fenster wird geöffnet, und die Zahl "45" wird im Textfeld angezeigt. Wenn innerhalb von 45 Sekunden keine Eingabe über die Maus oder über die Tastatur erfolgt, wird der Bildschirm ausgeblendet.

Beispiel 2:

```
1> BLANKER CX_POPUP=no
```

Das Programm "Blanker" wird aufgerufen. Wenn innerhalb eines Zeitraums von 60 Sekunden (Vorgabe) keine Eingabe erfolgt, wird der Bildschirm ausgeblendet. Das BLANKER-Fenster wird nicht geöffnet.

Kapitel 8 enthält ein weiteres Beispiel zur Verwendung des Befehls BLANKER.

ClickToFront

Bei Verwendung dieses Programms können Sie ein Fenster in den Vordergrund bringen, indem Sie es irgendwo doppelt anklicken.

Format CLICKTOFRONT [*CX_PRIORITY* <Priorität>] [*QUALIFIER* <Kennzeichner>]

Schablone CX_PRIORITY/N/K,QUALIFIER/K

Pfad Extras:Tools/Commodities

Von CLICKTOFRONT wird kein Fenster geöffnet. Die Argumente entsprechen den Merkmalen im Fenster "Information" des Piktogramms "ClickToFront".

Wenn Sie das Programm von einer Shell aus aufgerufen haben, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

CrossDOS

Mit diesem Programm stellen Sie die Textfilter- und Konvertierungsoptionen für CrossDOS-Geräte ein.

Format CROSSDOS [*CX_PRIORITY* <Priorität>] [*CX_POPKEY* <Taste>] [*CX_POPUP* <yes | no>]

Schablone CX_PRIORITY/N/K,CX_POPKEY/K,CX_POPUP/K,

Pfad Extras:Tools/Commodities

Mit CrossDOS können Sie über die Standard-Amiga-Laufwerke Daten von MS-DOS-formatierten Disks lesen und auf sie schreiben.

Wenn Sie das Programm von einer Shell aus aufgerufen haben, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

Exchange

Mit diesem Programm werden die Commodity-Exchange-Programme gesteuert und überwacht.

Format	EXCHANGE [*CX_PRIORITY* <Priorität>] [*CX_POPKEY*<Taste>] [*CX_POPUP* <yes | no>]
Schablone	CX_PRIORITY/N/K,CX_POPKEY/K,CX_POPUP/K
Pfad	Extras:Tools/Commodities

Wenn Sie das Programm von einer Shell aus aufgerufen haben, drücken Sie Ctrl-C, um das Programm zu verlassen.

Beispiel:

```
1> EXCHANGE CX_POPKEY "Shift F1"
```

Das Programm "Exchange" wird aufgerufen und das zugehörige Fenster angezeigt. Wenn das Fenster verdeckt ist, können Sie es durch Drücken der Tastenkombination Shift-F1 in den Vordergrund holen.

FKey

Mit diesem Programm können Sie Befehle bestimmten Tastenfolgen zuordnen und sich somit Eingabearbeit sparen.

Format	FKEY [*CX_PRIORITY* <Priorität>] [*CX_POPKEY* <Taste>] [*CX_POPUP* <yes | no>]
Schablone	CX_PRIORITY/N/K,CX_POPKEY/K,CX_POPUP/K
Pfad	Extras:Tools/Commodities

Mit FKEY können Sie acht Befehle einer beliebigen, einzugebenden Tastenfolge zuordnen.

Wenn Sie das Programm von einer Shell aus aufrufen, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

MouseBlanker

Dieses Programm dient zum Ausblenden des Mauspfeil, wenn Daten über die Tastatur eingegeben werden.

Format	MOUSEBLANKER [*CX_PRIORITY* <Priorität>]
Schablone	CX_PRIORITY/N/K
Pfad	Extras:Tools/Commodities

Wenn Sie das Programm von einer Shell aus aufrufen, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

NoCapsLock

Mit diesem Programm wird die Taste "Caps Lock" (Nur Großbuchstaben) inaktiviert.

Format	NOCAPSLOCK [*CX_PRIORITY*<Priorität>]
Schablone	CX_PRIORITY/N/K
Pfad	Extras:Tools/Commodities

Wenn Sie das Programm von einer Shell aus aufrufen, drücken Sie Ctrl-C oder verwenden Sie den Befehl BREAK, um das Programm zu verlassen.

7.3 Weitere Workbench-bezogene Hilfsprogramme und Programme

Mit der folgenden Gruppe von Befehlen rufen Sie Workbench-Hilfs- und Dienstprogramme auf. Weitere Informationen finden Sie im Workbench-Handbuch.

Calculator

Mit diesem Programm wird ein Taschenrechner auf dem Bildschirm angezeigt.

Format CALCULATOR [*PUBSCREEN* <Public-Schirm>] [*TAPE* <Fenster>]

Schablone PUBSCREEN,TAPE/K

Pfad Extras:Tools

Sie können die Ergebnisse des Taschenrechners mit der Maus kopieren und in Konsolenfenster (z. B. Shell- oder ED-Fenster) einfügen.

Mit der Option TAPE legen Sie die Größe des CALCULATOR-Fensters für Ein- und Ausgabedaten fest. Die Angabe muß in folgendem Format erfolgen:

```
TAPE=RAW:x/y/Breite/Höhe/Titel/Optionen
```

Eine detaillierte Beschreibung der Optionen und Argumente für das TAPE-Fenster können Sie der Beschreibung der Fensterspezifikation für den Befehl NEWSHELL in Kapitel 6 entnehmen.

Clock

Mit diesem Programm wird eine Uhr auf dem Bildschirm angezeigt.

Format	CLOCK [DIGITAL] [<LEFT>] [<TOP>] [<WIDTH>] [<HEIGHT>] [24HOUR] [SECONDS] [DATE] [<FORMAT><n>] [*PUBSCREEN* <Public-Schirm>]
Schablone	DIGITAL/S,LEFT/N,TOP/N,WIDTH/N, HEIGHT/N,24HOUR/S,SECONDS/S,DATE/S, FORMAT/N,PUBSCREEN/K
Pfad	SYS:Utilities

Bei Angabe der Option DIGITAL wird eine digitale Uhr angezeigt. Ansonsten erscheint eine analoge Uhr, deren Größe geändert werden kann (Vorgabe).

Mit den Optionen LEFT (links), TOP (oben), WIDTH (Breite) und HEIGHT (Höhe) können Sie die Größe und die Position der Uhr angeben. Die Schlüsselwörter müssen nicht eingegeben werden. Numerische Argumente werden aber wie folgt als Angaben zur Position der Uhr interpretiert:

Erste Zahl	Die Uhr wird <n> Bildpunkte vom linken Bildschirmrand entfernt geöffnet.
Zweite Zahl	Die Uhr wird <n> Bildpunkte vom oberen Bildschirmrand entfernt geöffnet.
Dritte Zahl	Die analoge Uhr ist <n> Bildpunkte breit.
Vierte Zahl	Die analoge Uhr ist <n> Bildpunkte hoch.

Wenn z. B. nur Breite und Höhe der Uhr angegeben werden soll, müssen Sie die Schlüsselwörter WIDTH und HEIGHT eingeben. Wenn Sie nur zwei Zahlen ohne die zugehörigen Schlüsselwörter eingeben, werden sie als die Optionen LEFT und TOP interpretiert. Die Optionen WIDTH und HEIGHT werden in Verbindung mit der Option DIGITAL ignoriert.

Bei Verwendung der Option 24HOUR wird die Uhrzeit im 24-Stunden-Format angezeigt. Diese Option steht bei der analogen Uhr nicht zur Verfügung.

Wenn Sie die Option SECONDS angeben, erscheint bei der analogen Uhr zusätzlich ein Sekundenzeiger. In Verbindung mit der Option DIGITAL hat diese Option keine Auswirkungen.

Über die Option DATE wird das Datum aufgerufen.

Die Option FORMAT bezieht sich nur auf die digitale Uhr. Für FORMAT kann ein Wert von 0 bis 5 angegeben werden. Diese Werte stehen für die sechs möglichen Digitalformate (siehe Menü auf der Workbench) . Die Formate 4 und 5 sind von der Einstellung im Landes-Voreinsteller (Locale) abhängig. Zur Angabe eines Digitalformats verwenden Sie entweder das Schlüsselwort FORMAT oder die Werte LEFT, TOP, WIDTH und HEIGHT. Die Werte WIDTH und HEIGHT dienen dabei nur als Platzhalter und werden ignoriert.

Beispiel 1:

Wenn eine Uhr geöffnet werden soll, die sich jeweils 75 Bildpunkte vom linken und vom oberen Bildschirmrand entfernt befindet sowie 300 Bildpunkte breit und 100 Bildpunkte hoch ist, geben Sie folgenden Befehl ein:

```
1> CLOCK 75 75 300 100
```

Beispiel 2:

Wenn die Sekunden und das Datum angezeigt werden sollen, geben Sie folgenden Befehl ein:

```
1> CLOCK SECONDS DATE
```

Beispiel 3:

Wenn eine digitale Uhr mit 24-Stunden-Format angezeigt werden soll, die sich 320 Bildpunkte vom linken Bildschirmrand und 0 Bildpunkte vom oberen Bildschirmrand entfernt (d. h. in der Titelleiste) befindet, geben Sie folgenden Befehl ein:

```
1> CLOCK DIGITAL 320 0 FORMAT 2
```

Kapitel 8 enthält weitere Beispiele zur Verwendung des Programms CLOCK (Uhr).

CMD

Mit diesem Programm werden Daten, die an eine serielle oder parallele Schnittstelle ausgegeben werden, in eine Datei umgeleitet.

Format	CMD <Gerätename> <Dateiname> [*OPT* S | M | N]
Schablone	DEVICENAME/A,FILENAME/A,OPT/K
Pfad	Extras:Tools

Für <Gerätename> kann "serial" oder "parallel" angegeben werden. Sollen Druckerausgabedaten umgeleitet werden, muß das Gerät den Angaben im Drucker-Voreinsteller (Printer) entsprechen. <Dateiname> steht für den Namen der Datei, in die die umgeleiteten Ausgabedaten geschrieben werden sollen.

Der Befehl CMD hat folgende Optionen:

S	Kurze Initialisierungssequenzen werden übersprungen (normalerweise ein Zurücksetzen, wenn ein Bildschirmausdruck umgeleitet wird).
M	Die Ausgabedaten für mehrere Dateien werden so lange umgeleitet, bis Sie den Befehl BREAK eingeben oder die Tastenkomibnation Ctrl-C drücken.
N	Dem Benutzer wird durch Meldungen der Status des Prozesses angezeigt.

Beispiel:

```
1> CMD parallel RAM:cmd_file
```

Mit diesem Befehl werden alle Ausgabedaten eines einzelnen Programmlaufs, die an die parallele Schnittstelle gesendet werden, in die Datei "cmd_file" im RAM: umgeleitet.

DiskCopy

Mit diesem Befehl wird der Inhalt einer Disk 1:1 auf eine andere kopiert.

Format	DISKCOPY [FROM] <Gerät> [TO] <Gerät> [*NAME* <Name>] [NOVERIFY] [MULTI]
Schablone	FROM/A,TO/A,NAME/K,NOVERIFY/S,MULTI/S,
Pfad	SYS:System

Mit den beiden Parametern <Gerät> werden die Laufwerke angegeben, von denen bzw. auf die kopiert wird (z. B. DF0: oder DF1:).

Standardmäßig hat die Zieldisk den gleichen Namen wie die Quelldisk. Mit der Option NAME können Sie der Zieldisk einen anderen Namen geben.

Normalerweise wird bei der Ausführung dieses Befehls jeder einzelne Datenzylinder kopiert und überprüft. Bei Angabe der Option NOVERIFY wird keine Überprüfung durchgeführt, wodurch der Kopiervorgang beschleunigt wird.

Bei Angabe der Option MULTI werden die Daten von der Quelldisk vollständig in den Speicher geladen. Dadurch können Sie mehrere Kopien anfertigen, ohne daß die Daten jedesmal von der Quelldisk eingelesen werden müssen.

Beispiel 1:

```
1> DISKCOPY DF0: TO DF2:
```

Mit diesem Befehl wird der Inhalt der Diskette in Laufwerk DF0: auf die Diskette in Laufwerk DF2: kopiert. Dabei wird der Inhalt der Diskette in Laufwerk DF2: überschrieben.

Beispiel 2:

```
1> DISKCOPY DF0: TO DF2: NAME NeueDisk NOVERIFY
```

Mit diesem Befehl wird der Inhalt der Diskette in Laufwerk DF0: auf die Diskette in Laufwerk DF2: kopiert. Diese Diskette erhält den Namen "NeueDisk". Die Diskette wird während des Kopiervorgangs nicht überprüft.

FixFonts

Mit diesem Befehl werden die Dateien mit der Erweiterung ".font" im Verzeichnis FONTS: aktualisiert.

Format FIXFONTS

Schablone (keine)

Pfad SYS:System

Vom Programm "FixFonts" wird kein Fenster geöffnet, und es werden keine Ausgabedaten generiert. Beim Aktualisieren des Verzeichnisses FONTS: leuchtet die Laufwerkskontrolleuchte. Wenn die Aktualisierung beendet ist, erlischt die Kontrolleuchte, und die Shell-Eingabeaufforderung wird wieder angezeigt. Ist die Disk, auf der das Verzeichnis FONTS: liegt und aktualisiert werden soll, nicht vorhanden oder tritt ein anderer Fehler auf, wird ein Standarddialogfenster zur Angabe des Fehlers angezeigt.

Verwenden Sie stets den Befehl FIXFONTS, nachdem Sie Veränderungen im Verzeichnis FONTS: vorgenommen haben, wie z. B. Kopieren neuer Zeichensatzdateien oder Löschen von einzelnen Zeichensatzgrößen.

Format

Mit diesem Befehl werden Disks für den Gebrauch auf dem Amiga formatiert.

Format FORMAT *DEVICE* | *DRIVE* <Gerät> *NAME* <Name>
[OFS | FFS]
[INTERNATIONAL | NOINTERNATIONAL]
[DIRCACHE | NODIRCACHE] [NOICONS] [QUICK]

Schablone DEVICE=DRIVE/K/A,NAME/K/A,OFS/S,FFS/S,
INTL=INTERNATIONAL/S,
NOINTL=NOINTERNATIONAL/S,DIRCACHE/S,
NODIRCACHE/S,NOICONS/S,QUICK/S

Pfad SYS:System

Zum Formatieren einer Disk müssen Sie die Schlüsselwörter DEVICE (oder DRIVE) und NAME angeben. Der Name kann maximal 31 Zeichen lang sein. Wenn der Name Leerzeichen enthält, müssen Sie den gesamten Namen in doppelte Anführungszeichen setzen.

Mit der Option OFS wird die Disk mit dem "Old File System" (altes Dateisystem) formatiert. Bei Angabe der Option FFS wird die Disk mit dem "Fast File System" (schnelles Dateisystem) formatiert. FFS-formatierte Disks ermöglichen eine schnellere Verarbeitung als OFS-formatierte Disks. FFS-formatierte Disks sind jedoch zu Amiga-Computern mit Systemsoftware vor Version 2.04 nicht kompatibel. Die Standardeinstellung für Disketten ist OFS, für PCMCIA-Karten und Festplattenpartitionen FFS.

Bei Angabe der Option INTERNATIONAL werden Disks mit der internationalen Version der Dateisysteme formatiert. Diese internationalen Dateisysteme unterscheiden auch bei Umlauten in den Dateinamen nicht zwischen Groß- und Kleinschreibung. Mit der Option NOINTERNATIONAL werden Disks, für die standardmäßig der internationale Modus verwendet wird, zwangsweise mit dem nichtinternationalen Dateisystem formatiert. Die Standardeinstellung für Disketten und PCMCIA-Karten ist NOINTERNATIONAL, für Festplattenpartitionen INTERNATIONAL. Disks, die mit dem Modus INTERNATIONAL formatiert werden, sind nicht zu Amiga-Computern mit Systemsoftware vor Version 2.04 kompatibel.

Über die Option DIRCACHE wird der Verzeichnis-Cache aktiviert. Dadurch wird das Öffnen von Schubladen, Dateien, Dialogfenstern und Auflistungen beschleunigt. Wenn Sie den Verzeichnis-Cache für Disketten und Systeme mit langsamen Festplatten verwenden, wird das Auflisten von Verzeichnissen und das Öffnen von Workbench-Fenstern beschleunigt. Auf Systemen mit schnellen Festplatten ist die Verwendung des Verzeichnis-Cache nicht sinnvoll. Mit der Option NODIRCACHE wird der Verzeichnis-Cache inaktiviert. Disks, die mit Verzeichnis-Cache formatiert werden, sind zu Amiga-Computern mit Systemsoftware vor Version 3.0 nicht kompatibel. Die Standardeinstellung für Disketten und PCMCIA-Karten ist DIRCACHE, für Festplatten NODIRCACHE.

Mit der Option NOICONS wird verhindert, daß ein Papierkorbpiktogramm mit zugehörigem Verzeichnis auf der neuformatierten Disk eingerichtet wird.

Bei Angabe der Option QUICK werden nur der Root-Block (mit Spur) und der Boot-Block (mit Spur) formatiert und erstellt und die Bitmap-Blöcke angelegt. Diese Option können Sie nur für bereits formatierte Disks verwenden. Mit dieser Option können Sie also keine mit CrossDOS-formatierten Disketten für AmigaDOS neu formatieren (oder umgekehrt).

Beispiel 1:

```
1> FORMAT DRIVE DF0: NAME LeereDiskette
```

Mit diesem Befehl wird die Diskette in Laufwerk DF0: formatiert. Dabei werden eventuell vorhandene Daten gelöscht, und die Diskette erhält den Namen "LeereDiskette".

Beispiel 2:

```
1> FORMAT DRIVE DF2: NAME NeueDiskette QUICK
```

Eine Diskette, die bereits Daten enthält, wird neu formatiert, d. h. die Daten werden gelöscht.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls FORMAT.

GraphicDump

Mit diesem Befehl wird der im Vordergrund befindliche Bildschirm gedruckt.

Format	GRAPHICDUMP [TINY | SMALL | MEDIUM | LARGE | <xPunkte>:<yPunkte>]
Schablone	TINY/S,SMALL/S,MEDIUM/S,LARGE/S, <xPunkte>:<yPunkte>/S
Pfad	Extras:Tools

Mit dem Befehl GRAPHICDUMP wird nach zehn Sekunden der im Vordergrund befindliche Bildschirm gedruckt. Innerhalb dieser Frist kann der gewünschte Bildschirm in den Vordergrund geholt werden.

Die Größenoptionen, die den gültigen Merkmalen (Tool Types) des Programms entsprechen, bestimmen die Breite des Ausdrucks:

TINY	1/4 der auf dem Drucker möglichen Gesamtbreite
SMALL	1/2 der auf dem Drucker möglichen Gesamtbreite
MEDIUM	3/4 der auf dem Drucker möglichen Gesamtbreite
LARGE	die volle auf dem Drucker mögliche Gesamtbreite

Die Höhe des Ausdrucks wird entsprechend dem Seitenverhältnis Breite:Höhe des Bildschirms eingestellt.

Zur Angabe der genauen Abmessungen geben Sie die absoluten Werte für die Breite (<xPunkte>) und für die Höhe (<yPunkte>) mit den Druckerpunkten als Maßeinheit ein. Die beiden Werte müssen durch einen Doppelpunkt (:) getrennt sein.

Beispiel 1:

```
1> GRAPHICDUMP SMALL
```

Mit diesem Befehl wird der Bildschirm im Vordergrund ausgedruckt. Er ist ungefähr halb so breit wie die auf dem Drucker mögliche Gesamtbreite.

Beispiel 2:

```
1> GRAPHICDUMP 600:300
```

Bei diesem Befehl ist der Ausdruck 600 Punkte hoch und 300 Punkte breit.

IconEdit

Mit diesem Befehl wird das Aussehen und der Typ von Piktogrammen geändert. Über den Befehl ICONEDIT wird das Programm "IconEdit" geöffnet. Es dürfen keine Argumente eingegeben werden. Detaillierte Informationen zum Programm "IconEdit" können Sie im *Workbench-Benutzerhandbuch* nachlesen.

InitPrinter

Mit diesem Befehl wird ein Drucker auf die Druckoptionen initialisiert, die im Voreinsteller-Editor angegeben wurden.

Format	INITPRINTER
Schablone	(keine)
Pfad	Extras:Tools

Nach Ausführung des Programms "InitPrinter" und Zurücksetzen des Druckers wird wieder die Shell-Eingabeaufforderung angezeigt. Der Drucker wird automatisch beim ersten Zugriff initialisiert. Wenn Sie jedoch den Drucker zwischendurch ausschalten, müssen Sie den Drucker ggf. mit dem Programm "InitPrinter" erneut initialisieren.

Intellifont

Mit diesem Befehl wird das Programm "Intellifont" zur Verwaltung von Umriß-Zeichensätzen aufgerufen.

Format	INTELLIFONT [VALIDATE]
Schablone	VALIDATE/S
Pfad	SYS:System

Mit dem Programm "Intellifont" werden neue Umriß-Zeichensätze auf Ihrem System installiert. Außerdem können Sie die Größe vorhandener Zeichensätze ändern und nicht mehr benötigte Zeichensätze löschen. Darüber hinaus können Sie Bitmap-Versionen beliebiger Größe für Anwendungen erstellen, die ansonsten solche Zeichensätze nicht unterstützen.

Bei Angabe der Option VALIDATE wird überprüft, ob die Zeichensätze ordnungsgemäß installiert wurden und alles zu deren Darstellung Erforderliche vorhanden ist.

KeyShow

Mit diesem Befehl wird die aktuelle Tastaturbelegung angezeigt. Für den Befehl KEYSHOW dürfen keine Argumente angegeben werden.

Detaillierte Informationen zum Programm "KeyShow" können Sie im *Workbench-Benutzerhandbuch* nachlesen.

MEmacs

Mit diesem Befehl wird ein bildschirmorientierter Texteditor geöffnet.

Format	MEMACS [<Dateiname>] [*OPT* W] [*GOTO* <n>]
Schablone	FROM/M,OPT/K,GOTO/K
Pfad	Extras:Tools

Der Texteditor "MEmacs" wird in Kapitel 4 dieses Handbuchs beschrieben.

More

Mit diesem Befehl wird der Inhalt der angegebenen ASCII-Datei im Shell-Fenster angezeigt. Dem Programm "More" ist kein Piktogramm zugeordnet. Das Programm wurde inzwischen durch das Programm "MultiView" ersetzt, aber es ist weiterhin verfügbar.

Format	MORE <Dateiname>
Schablone	(keine)
Pfad	SYS:Utilities

Wenn sich die Datei nicht im aktuellen Verzeichnis befindet, müssen Sie den vollständigen Pfad angeben. Wenn Sie keine Datei angeben, wird ein Dateiauswahlfenster angezeigt.

Wenn Sie die Taste H drücken, werden Erläuterungen zu den Befehlstasten innerhalb des Programms "More" angezeigt. Folgende Befehle können innerhalb "More" verwendet werden:

<Leerzeichen>	Nächste Seite (More).
<Eingabetaste>	Nächste Zeile.
q oder Ctrl-C	Verlassen.
h	Hilfe.

/Zeichenkette	Nach Zeichenkette suchen (unter Berücksichtigung von Groß und Kleinschreibung). Mit der Taste n wird später nach der nächsten übereinstimmenden Zeichenkette gesucht.
.Zeichenkette	Nach Zeichenkette suchen (ohne Berücksichtigung von Groß- und Kleinschreibung).
n	Nächste übereinstimmende Zeichenkette suchen.
Ctrl-L	Fenster aktualisieren.
<	Erste Seite.
>	Letzte Seite.
%N	Bis N% in die Datei hineingehen.
b oder <Rücktaste>	Vorherige Seite (niedrigere Seitennummer).
E	Aktuelle Datei mit dem in ENV:EDITOR eingestellten Editor edieren.

Von MORE werden auch Eingaben über den Befehl PIPE akzeptiert. Da die Eingabe über den Befehl PIPE keine feste Länge hat, werden die Befehle für "Vorherige Seite" (Rücktaste oder b), "Letzte Seite" (>) und "N% in die Datei hinein" (N%) inaktiviert.

Wenn Sie das Programm von der Shell aus aufrufen und die Variable EDITOR definiert wurde, können Sie durch Drücken der Tastenkombination Shift-E einen Editor öffnen, um die angezeigte Datei zu edieren. In der Variable EDITOR muß der vollständige Pfad zum gewünschten Editor angegeben werden (z. B. C:ED).

Beispiel:

```
1> MORE DF0:Testdatei
```

Mit diesem Befehl wird der Inhalt der ASCII-Datei "Testdatei" auf der Diskette in Laufwerk DF0: angezeigt.

MultiView

Mit diesem Programm werden Grafik-, Text-, AmigaGuide-Hilfs-, Sound- und Animationsdateien angezeigt.

Format MULTIVIEW [FILE <Dateiname>] [CLIPBOARD] [*CLIPUNIT* <Zwischenspeichereinheit>] [SCREEN] [*PUBSCREEN* <Public-Schirm>] [REQUESTER] [BOOKMARK] [*FONTNAME* <Zeichensatzname>] [*FONTSIZE* <Zeichengröße>] [BACKDROP] [WINDOW] [*PORTNAME* <ARexx-Schnittstellenname>]

Schablone FILE,CLIPBOARD/S,CLIPUNIT/K/N,SCREEN/S, PUBSCREEN/K,REQUESTER/S,BOOKMARK/S, FONTNAME/K,FONTSIZE/K/N,BACKDROP/S, WINDOW/S,PORTNAME/K

Pfad SYS:Utilities

Wenn sich die anzuzeigende Datei nicht im aktuellen Verzeichnis befindet, müssen Sie den vollständigen Pfad angeben. Geben Sie keinen Dateinamen an, wird ein Dateiauswahlfenster angezeigt.

Wenn Sie CLIPBOARD angeben, wird anstelle einer Datei der Inhalt des Zwischenspeichers angezeigt. CLIPUNIT steht für die Speichereinheit, die bei Angabe des Schlüsselworts CLIPBOARD verwendet werden soll.

SCREEN gibt an, daß ein Objekt auf einem eigenen Bildschirm erscheinen soll, wobei der im Objekt angegebene Bildschirmmodus verwendet wird. Wurde z. B. für eine ILBM-Bilddatei "Low Res" (niedrige Auflösung) angegeben, wird von MultiView ein entsprechender Bildschirm geöffnet.

Bei Angabe von REQUESTER wird ein Dateiauswahlfenster angezeigt.

Über BOOKMARK wird ein Objekt mit dessen zugehöriger Position aufgerufen, wenn eine Datei mit dieser Markierung geöffnet wird.

FONTNAME gibt den Zeichensatz an, der bei Anzeige eines Textobjekts verwendet werden soll. FONTSIZE steht dabei für die Zeichengröße in Punkt.

BACKDROP gibt an, daß das Fenster als Hintergrundfenster erscheinen soll.

Über WINDOW kann ein MultiView-Fenster geöffnet werden, ohne daß eine Aufforderung zum Laden einer Datei erfolgt. Auf diese Weise können Sie stets ein kleines MultiView-Fenster auf dem Workbench-Bildschirm geöffnet haben, in das Sie bei Bedarf Piktogramme ziehen können.

Über PORTNAME können Sie einen ARexx-Schnittstellennamen für die Kommunikation mit "MultiView" angeben. Falls keine Angabe erfolgt, erhält das Fenster den Standardschnittstellennamen "MultiView.x". Dabei steht x für eine fortlaufende Schnittstellennummer (beginnend mit 1). (Beispiel: Wenn drei MultiView-Anzeigefenster geöffnet sind, lauten deren Schnittstellennamen MultiView.1, MultiView.2 und MultiView.3.) Mit Hilfe dieses Schnittstellennamens können Sie innerhalb von ARexx-Skripts auf bestimmte MultiView-Anzeigefenster verweisen.

Von "MultiView" werden die folgenden ARexx-Befehle unterstützt:

OPEN Ein Objekt aus der angegebenen Datei oder der Zwischenspeichereinheit wird geöffnet. Folgende Optionen sind möglich:

FILENAME/K Gibt den Namen der Objektdatei an.

CLIPBOARD/S Gibt an, daß das Objekt sich im Zwischenspeicher befindet.

CLIPUNIT/K/N Gibt die Zwischenspeichereinheit an, aus der das Objekt bei Angabe der Option CLIPBOARD abzurufen ist.

RELOAD Das aktuelle Objekt wird erneut geladen.

SAVEAS Das Objekt wird in der angegebenen Datei gespeichert. Wurde ein Block markiert, wird nur dieser Block gespeichert. Für SAVEAS gibt es nur das Argument NAME/K.

PRINT Der aktuelle Inhalt wird gedruckt. Ist ein Block markiert, wird nur dieser Block gedruckt.

ABOUT Das Dialogfenster "Version, Copyright" wird aufgerufen.

QUIT	MultiView wird geschlossen.
COPY	Der aktuelle Inhalt wird in den Zwischenspeicher kopiert. Ist ein Block markiert, wird nur dieser Block kopiert.
CLEARSELECTED	Der markierte Block wird gelöscht.
GETTRIGGERINFO	Die Befehle für die Auslösemethoden (engl. trigger methods) des aktuellen Objekts werden angezeigt. Die Optionen sind VAR/S und STEM/K.

Bei Angabe der Option STEM werden folgende stem-Erweiterungen verwendet:

.COUNT	Anzahl der Elemente
.n.LABEL	Marke
.n.COMMAND	Befehl
.n.METHOD	Numerische Methode

DOTRIGGERMETHOD	Die Auslösemethode wird für das aktuelle Objekt ausgeführt. Die Option ist METHOD/A.
SCREEN	Mit diesem Befehl wird angegeben, ob ein Objekt auf einem eigenen Schirm angezeigt werden soll. Die Optionen sind TRUE/S und FALSE/S.
PUBSCREEN	Mit diesem Befehl wird der Name des Public-Schirms angegeben, auf dem das Objekt angezeigt werden soll. Die Option ist NAME/A.
GETCURRENTDIR	Der vollständige Name des aktuellen Verzeichnisses, das dem aktuellen Objekt zugeordnet ist, wird angezeigt.
GETFILEINFO	Der vollständige Pfad und der Dateiname des aktuellen Objekts werden angezeigt.

GETOBJECTINFO	Der Name, der Basisname, die Gruppe und die ID des aktuellen Objekts werden angezeigt. Die Optionen sind VAR/S und STEM/K.
	Bei Angabe der Option STEM werden folgende stem-Erweiterungen verwendet:
	.FILENAME — Dateiname des Objekts
	.NAME — Beschreibender Name des Datentyps (DataType)
	.BASENAME — Basisname des Datentyps
	.GROUP — Gruppe, die den Datentyp enthält
	.ID — ID-Zeichenkette für den Datentyp
MINIMUMSIZE	Das Fenster wird in Abhängigkeit vom Typ des Inhalts in minimaler Größe dargestellt.
NORMALSIZE	Das Fenster wird in Abhängigkeit vom Typ des Inhalts in Normalgröße dargestellt.
MAXIMUMSIZE	Das Fenster wird in Abhängigkeit vom Typ des Inhalts in maximaler Größe dargestellt.
WINDOWTOFRONT	Das Fenster wird in den Vordergrund des Bildschirms geholt.
WINDOWTOBACK	Das Fenster wird in den Hintergrund des Bildschirms gestellt.
SCREENTOFRONT	Der Bildschirm wird in den Vordergrund der Anzeige geholt.
SCREENTOBACK	Der Bildschirm wird in den Hintergrund der Anzeige gestellt.
ACTIVATEWINDOW	Das Fenster wird aktiviert.
BEEPSCREEN	Ein akustisches Signal (Piepsen) wird auf dem Bildschirm ausgelöst, in dem sich das Fenster befindet.

NoFastMem

Mit diesem Befehl wird der Amiga gezwungen, nur das Chip-RAM zu verwenden. Über NOFASTMEM wird das vom System benutzte FAST-RAM (oder RAM-Erweiterungen) inaktiviert. Soll das FAST-RAM wieder aktiviert werden, unterbrechen Sie das Programm "NoFastMem" durch Eingabe des Befehls BREAK oder durch

Drücken der Tastenkombination Ctrl-C oder durch nochmaligen Aufruf aus einer anderen Shell. NOFASTMEM wird ohne Optionen verwendet. Detaillierte Informationen zum Progamm "NoFastMem" können Sie dem *Workbench-Benutzerhandbuch* entnehmen.

PrepCard

Mit diesem Befehl wird eine PCMCIA-Speicherkarte für die Verwendung als Disk-Gerät oder System-RAM in Systemen mit Kartensteckplätzen vorbereitet.

Format	PREPCARD [DISK|RAM]
Schablone	DISK/S,RAM/S
Pfad	Extras:Tools

Über die Option DISK wird die PCMCIA-Karte zur Verwendung als Disk-Gerät CC0: vorbereitet. Die Schreib-/Lesevorgänge für mit DISK vorbereitete Karten entsprechen den Schreib-/Lesevorgängen für Disketten. Es empiehlt sich die Verwendung einer batteriegepufferten PCMCIA-Karte als Disk, wenn Ihre Arbeit nicht verloren gehen soll.

Über die Option RAM wird die PCMCIA-Karte zur Verwendung als System-RAM vorbereitet. Bei einem Start oder Neustart des Systems mit eingesteckter Speicherkarte wird die Speicherkapazität der Karte zur vorhandenen Speicherkapazität hinzugefügt. Mit Ausnahme von ROM-Karten (ROM - Read-Only Memory) können Sie beliebige PCMCIA-Karten als RAM verwenden.

Die Optionen DISK und RAM schließen einander aus. Wenn Sie beide angeben, wird vom Programm "PrepCard" nur die erste angegebene Option berücksichtigt und die zweite ignoriert. Wenn Sie das Programm "PrepCard" auf einem System zu starten versuchen, das nicht mit Steckplätzen ausgestattet ist, wird eine entsprechende Fehlermeldung angezeigt.

Kapitel 8

Befehlsbeispiele

Die Befehlsbeispiele in anderen Kapiteln dieses Handbuchs dienen in erster Linie zur Erläuterung der korrekten Syntax sowie der allgemeinen Arbeitsweise von AmigaDOS. In diesem Kapitel werden die Befehle beschrieben, die zur Ausführung einer großen Anzahl allgemeiner Aufgaben nützlich sind.

Dieses Kapitel ist in drei Abschnitte untergliedert:

- Grundlegende Aufgaben
- Gelegentliche Aufgaben
- Komplexere Aufgaben

8.1 Grundlegende Aufgaben

Dieser Abschnitt richtet sich insbesondere an unerfahrene Shell-Benutzer und behandelt Befehle und kurze Befehlsdateien, die zur Ausführung grundlegender Aufgaben erforderlich sind. Verwenden Sie die Befehle als Modelle für Ihre eigenen Befehle und ersetzen Sie dabei die Namen von Disks, Verzeichnissen und Dateien. Zur Verwendung eines Befehls geben Sie nur die hinter der Eingabeaufforderung (in der Regel `1>`) angegebenen Daten ein. Anschließend drücken Sie zum Ausführen der jeweiligen Befehlszeile die Eingabetaste.

8.1.1 Öffnen eines Shell-Fensters

Zum Öffnen eines Shell-Fensters von der Workbench aus gehen Sie wie folgt vor:

1. Öffnen Sie die Schublade "System" auf Ihrer Workbench-Disk bzw. -Partition.
2. Klicken Sie das Shell-Piktogramm doppelt an.

   ODER

1. Wählen Sie im Workbench-Menü den Menüpunkt "Befehl ausführen..." aus.
2. Im daraufhin angezeigten Dialogfenster geben Sie den Befehl **NEWSHELL** ein.

Zum Öffnen eines weiteren Shell-Fensters von einer Shell aus geben Sie den Befehl NEWSHELL hinter einer Shell-Eingabeaufforderung ein:

```
1> NEWSHELL
```

8.1.2 Starten von Programmen von der Shell aus

Zum Starten eines Programms, das sich im Suchpfad befindet, geben Sie den Namen des Programms hinter der Eingabeaufforderung ein:

```
1> Clock
```

Zum Starten eines Programms, das sich nicht im Suchpfad befindet, geben Sie den vollständigen Pfad zum jeweiligen Programm ein:

```
1> Tempus:Fugit/Utils/SuperClock
```

Zum Starten eines Programms, das sich zwar nicht im Suchpfad befindet, aber in einem Unterverzeichnis des aktuellen Verzeichnisses, geben Sie den relativen Pfad zum jeweiligen Programm ein:

```
1> Utils/SuperClock
```

8.1.3 Stoppen eines Programms

AmigaDOS-Befehle und der Großteil der Workbench-Programme, die von der Shell aus gestartet wurden und zur Zeit ausgeführt werden, können durch Drücken der Tastenkomination Ctrl-C verlassen bzw. gestoppt werden. Diese Möglichkeit ist besonders wichtig, wenn ein Löschvorgang (DELETE) mit Namensmuster gestoppt werden muß oder Verzeichnisauflistungen oder andere zeitintensive Prozesse unterbrochen werden sollen. Befehlsdateien (Skripts) können mit der Tastenkombination Ctrl-D gestoppt werden.

Zum Stoppen der Ausführung eines Befehls bzw. eines Programms gehen Sie wie folgt vor:

1. Klicken Sie das Shell-Fenster an, von dem aus der Befehl bzw. das Programm gestartet wurde. Dadurch wird es zum aktuellen Fenster.
2. Drücken Sie die Tastenkombination Ctrl-C.

In einigen Fällen müssen Sie anschließend die Eingabetaste drücken, damit die Shell-Eingabeaufforderung wieder angezeigt wird.

Zum Stoppen der Ausführung einer Befehlsdatei gehen Sie wie folgt vor:

1. Klicken Sie das Shell-Fenster an, von dem aus die Befehlsdatei gestartet wurde. Dadurch wird es zum aktuellen Fenster.
2. Drücken Sie die Tastenkombination Ctrl-D.

8.1.4 Wechseln des aktuellen Verzeichnisses

Das aktuelle Verzeichnis wird in der Regel als Bestandteil der vorgegebenen Shell-Eingabeaufforderung angezeigt (z. B. `1.Workbench:>`). In den folgenden Beispielen können Sie anhand der Eingabeaufforderungen den Wechsel des aktuellen Verzeichnisses verfolgen.

Zum Einsparen von Eingabearbeit machen Sie stets das Verzeichnis, in dem Sie gerade arbeiten, zum aktuellen Verzeichnis.

Wenn mehrere Befehle eingegeben werden sollen, die sich auf ein bestimmtes Verzeichnis beziehen, wechseln Sie zunächst mit dem Befehl CD in dieses Verzeichnis. Mit den folgenden beiden Befehlsfolgen wird derselbe Task ausgeführt:

```
1.Arbeit:> COPY Storage/Keymaps/usa2 TO
  DEVS:Keymaps
1.Arbeit:> DELETE Storage/Keymaps/usa2

1.Arbeit:> CD Storage/Keymaps
1.Storage:Keymaps> COPY usa2 TO DEVS:Keymaps
1.Storage:Keymaps> DELETE usa2
```

Wenn Sie die zweite Befehlsfolge verwenden, sparen Sie über ein Dutzend Tastenanschläge ein. Diese Arbeitsersparnis wird noch größer, wenn Sie weitere Arbeitsschritte im Verzeichnis "Storage /Keymaps" (Speicher/Tastaturbelegungen) ausführen müssen.

Zum Wechseln des aktuellen Verzeichnisses mit dem geringsten Eingabeaufwand lassen Sie das Befehlswort CD aus. Verwenden Sie den Schrägstrich und den Doppelpunkt, um die Verzeichnisse innerhalb der Verzeichnisstruktur zu wechseln:

```
1.Workbench:Devs/Monitors> /Printers
1.Workbench:Devs/Printers> :Prefs/Presets
1.Workbench:Prefs/Presets> /
1.Workbench:Prefs>
```

Zum schnellen Hinundherwechseln zwischen zwei Verzeichnissen verwenden Sie die PCD-Befehlsdatei (im Verzeichnis S:)

```
1.Workbench:> PCD Devs/DOSDrivers
1.Workbench:Devs/DOSDrivers> Extras:Storage
1.Extras:Storage> PCD
1.Workbench:>
```

Zum Abfragen des aktuellen Verzeichnisses, wenn es nicht in der Shell-Eingabeaufforderung angezeigt wird, geben Sie den Befehl CD ohne Optionen ein:

```
1> CD
Workbench:
```

8.1.5 Ändern des Suchpfads

Ein Verzeichnis wird wie folgt auf dem Datenträger SYS: erstellt und dem Suchpfad für die aktuelle Shell hinzugefügt:

```
1> MAKEDIR SYS:MeinBefehl
1> PATH SYS:MeinBefehl ADD
```

Zum Hinzufügen des Verzeichnisses "MeinBefehl" in den Suchpfad, der für das gesamte System gilt, verwenden Sie anstelle von PATH den Befehl ASSIGN:

```
1> ASSIGN C: SYS:MeinBefehl ADD
```

Soll der Amiga im Verzeichnis C auf jeder beliebigen Disk in Laufwerk DF0: nach Befehlen suchen, verwenden Sie den Befehl ASSIGN mit der Option PATH:

```
1> ASSIGN C: DF0:C PATH
```

8.1.6 Anzeigen des Verzeichnisinhalts

Die Namen der Dateien und Unterverzeichnisse in einem Verzeichnis können Sie wie folgt mit dem Befehl DIR abfragen:

```
1> DIR DEVS:
     DataTypes (dir)
     Monitors (dir)
     DOSDrivers (dir)
     Keymaps (dir)
     Printers (dir)
  clipboard.device          DataTypes.info
  DOSDrivers.info           Keymaps.info
  mfm.device                Monitors.info
  parallel.device           postscript_init_ps
  printer.device            Printers.info
  serial.device             system-configuration
```

Zum Anzeigen der Namen der Dateien und Unterverzeichnisse sowie aller Dateien in den Unterverzeichnissen eines Verzeichnisses verwenden Sie zusätzlich das Schlüsselwort ALL (das Beispiel zeigt einen Auszug der Ausgabedaten):

```
1> DIR DEVS: ALL
  DataTypes (dir)
     8SVX                  8SVX.info
```

```
    AmigaGuide          AmigaGuide.info
    ANIM                ANIM.info
    CDXL                CDXL.info
    FTXT                FTXT.info
    ILBM                ILBM.info
  Monitors (dir)
    A2024               A2024.info
```

Sollen nur die Namen der Dateien (keine Verzeichnisse) angezeigt werden, verwenden Sie das Schlüsselwort FILES:

```
1> DIR DEVS: FILES
  clipboard.device          DataTypes.info
  DOSDrivers.info           Keymaps.info
  mfm.device                Monitors.info
  parallel.device           postscript_init_ps
  printer.device            Printers.info
  serial.device             system-configuration
```

Sollen nur die Namen der Dateien mit Ausnahme von .info-Dateien angezeigt werden, verwenden Sie Namensmuster:

```
1> DIR DEVS:~(#?.info) FILES
  clipboard.device          mfm.device
  parallel.device           postscript_init_ps
  printer.device            serial.device
  system-configuration
```

Sollen die Informationen über Dateien einschließlich Größe und Schutzbits aber ohne Datum und Uhrzeit angezeigt werden, verwenden Sie den Befehl LIST mit den Schlüsselwörtern FILES und NODATES:

```
1> LIST DEVS:~(#?.info) FILES NODATES
clipboard.device             6944 ----rw-d
mfm.device                   6684 ----rw-d
parallel.device              4272 ----rw-d
postscript_init_ps           5014 ----rw-d
printer.device              27420 ----rw-d
serial.device                5412 ----rw-d
system-configuration          232 ----rw-d
```

Sollen nur Informationen über eine einzelne Datei abgefragt werden, verwenden Sie den Befehl LIST mit dem Pfad zur gewünschten Datei:

```
1> LIST S:Startup-sequence
Directory "S:" on Tuesday 01-Dec-92
Startup-sequence  1360 -s--rw-d 30-Oct-92 12:00:21
1 file - 4 blocks used
```

Soll der durch ein Verzeichnis und dessen Inhalt belegte Speicherplatz einschließlich der Dateien in den Unterverzeichnissen angezeigt werden, verwenden Sie das Schlüsselwort ALL:

```
1> LIST ALL
```

Daraufhin wird zunächst der Inhalt des aktuellen Verzeichnisses angezeigt. Auf diese Auflistung folgt eine zusammenfassende Zeile. Beispiel:

```
TOTAL: 113 files - 762 blocks used

       (113 Dateien - 762 Blöcke belegt)
```

Wenn Sie die Anzahl der Blöcke durch 2 teilen, erhalten Sie die Speicherkapazität in KB.

Die Ausgabedaten der Befehle LIST und DIR, die aus dem Shell-Fenster gerollt sind, können Sie wie folgt abrufen:

Klicken Sie das Zoom-Symbol des Shell-Fensters einmal an. Dadurch wird zur alternativen Größe des Fensters gewechselt (in der Regel wird der gesamte Bildschirm ausgefüllt). Daraufhin wird zusätzlich der Teil des vorherigen Fensterinhalts angezeigt, der jetzt in das Fenster paßt. Wenn Sie das Zoom-Symbol erneut anklicken, wird das Fenster wieder in der vorherigen Größe angezeigt.

Wenn die maximale Bildschirmhöhe nicht zur vollständigen Anzeige der gewünschten Ausgabedaten reicht, geben Sie den Befehl erneut ein. Drücken Sie dazu die Taste "Cursor aufwärts" und die Eingabetaste. Anschließend unterbrechen Sie das Rollen der Bildschirmanzeige durch Drücken der Leertaste bzw. starten es wieder mit der Rücktaste.

Die Befehle CD und DIR können Sie wie folgt miteinander kombinieren:

Erstellen Sie die folgende Befehlsdatei und speichern Sie sie als "S:CDD". (Ein Beispiel zur Erstellung einer Befehlsdatei finden Sie in Kapitel 8.1.8 "*Erstellen einer Datei "User-startup*" auf Seite 8-10.)

```
.KEY dirpath
CD <dirpath>
DIR
```

Geben Sie **PROTECT S:CDD +s** ein, um das Schutzbit "s" der Befehlsdatei zu aktivieren. Wenn Sie später den Befehl CDD gefolgt von einem Pfad zu einem Verzeichnis eingeben, wird aufgrund der Befehlsdatei das angegebene Verzeichnis zum aktuellen Verzeichnis gemacht und gleich wird der Inhalt des Verzeichnisses angezeigt.

8.1.7 Kopieren von Dateien und Verzeichnissen

Wenn Sie eine einzelne Datei von einem Verzeichnis in ein anderes kopieren, müssen Sie nur die Pfade für die beiden Verzeichnisse eingeben, die Schlüsselwörter FROM und TO sind nicht erforderlich. Zur klareren Darstellung ist in den Beispielen dieses Handbuchs das Schlüsselwort TO meistens genannt, das Schlüsselwort FROM nicht.

Zum Kopieren einer Datei (unter Verwendung der wahlfreien Schlüsselwörter FROM und TO) in ein vorhandenes Verzeichnis namens Bilder geben Sie folgendes ein:

```
1> COPY FROM DF0:Pix/Fractal3 TO Arbeit:Bilder
```

Eine Datei wird wie folgt ohne wahlfreie Schlüsselwörter kopiert:

```
1> COPY DF0:Pix/Fractal3 Arbeit:Bilder
```

Zum Kopieren und gleichzeitigen Umbenennen einer Datei geben Sie den neuen Namen im Argument TO ein:

```
1> COPY DF0:Pix/Fractal3 TO Arbeit:Bilder/BestPic
```

Zum Kopieren aller Dateien aus einem Verzeichnis in ein anderes Verzeichnis, ohne das Verzeichnis selbst bzw. die darin enhaltenen Unterverzeichnisse zu kopieren, geben Sie folgendes ein:

```
1> COPY DF0:Pix TO Arbeit:Bilder
```

Der Inhalt des Verzeichnisses "DF0:Pix" wird im Verzeichnis "Arbeit:Bilder" abgelegt und nicht in einem eigenen Verzeichnis gruppiert. Wenn das Verzeichnis "Pix" Unterverzeichnisse enthält, werden diese nicht mitkopiert, sondern nur die .info-Dateien für möglicherweise im Verzeichnis "Pix" enthaltene Schubladen (Verzeichnisse). Dadurch entsteht zunächst der Eindruck, daß auch die Schubladen mitkopiert wurden.

Sollen alle Dateien von einem Verzeichnis in ein anderes kopiert und dabei das Verzeichnis selbst ohne darin enthaltene Unterverzeichnisse mitkopiert werden, geben Sie folgendes ein:

```
1> COPY DF0:Pix TO Arbeit:Bilder/Pix
```

Wenn es nicht bereits vorhanden ist, wird das Verzeichnis "Pix" im Verzeichnis "Arbeit:Bilder" angelegt. Das Zielverzeichnis muß nicht denselben Namen erhalten wie das Quellverzeichnis. Als Zielverzeichnis können Sie z. B. "Arbeit:Bilder/Fractale" angeben.

Soll das vollständige Verzeichnis einschließlich des Inhalts seiner Unterverzeichnisse in ein anderes Verzeichnis kopiert werden, verwenden Sie das Schlüsselwort ALL:

```
1> COPY DF0:Pix TO Arbeit:Bilder/Pix ALL
```

Das Verzeichnis "Arbeit:Bilder/Pix" ist danach ein Duplikat des Verzeichnisses "DF0:Pix".

Sollen nur bestimmte Dateien in ein anderes Verzeichnis kopiert werden, geben Sie bei ähnlichen Namen Namensmuster an:

```
1> COPY DF0:Pix/Fractal[3-7] TO Arbeit:Bilder/Pix
```

Die Dateien im Verzeichnis "DF0:Pix", deren Name mit "Fractal" beginnt und auf die Ziffern 3 bis 7 endet, werden kopiert.

Sollen spezifische Dateien in ein anderes Verzeichnis kopiert werden, geben Sie die Namen aller gewünschten Dateien ein. Wechseln Sie zunächst in das Quellverzeichnis, damit Sie nicht bei jedem Namen den vollständigen Pfad eingeben müssen:

```
1> CD DF0:Pix
1> COPY Fractal3 Julia Dragon TO Arbeit:Bilder/Pix
```

Wenn mehrere Dateien ohne Angabe des Schlüsselworts TO gleichzeitig kopiert werden, wird davon ausgegangen, daß es sich beim letzten Namen um den Namen des Zielverzeichnisses handelt. Während des Kopiervorgangs (Kopieren eines vollständigen Verzeichnisses, mehrerer Dateien oder mit Namensmuster) werden die Namen der kopierten Dateien und Verzeichnisse angezeigt.

8.1.8 Erstellen einer Datei "User-startup"

Die Datei "User-startup" (Benutzerstart) ist die Shell-Entsprechung der Workbench-Schublade "WBStartup". Dabei handelt es sich um eine Textdatei, die von der ursprünglichen Startsequenz (engl. Startup-sequence) aus wie eine Befehlsdatei aufgerufen wird. Schreiben Sie alle Konfigurationsbefehle, z. B. ASSIGN-Befehle, sowie die Namen aller Programme, die beim Systemstart ausgeführt werden sollen, in diese Datei.

Zum Erstellen einer Datei "User-startup" geben Sie folgendes ein:

```
1> RUN ED S:User-startup
```

Nach Öffnen des ED-Fensters geben Sie die gewünschten Befehle in aufeinanderfolgenden Zeilen ein. Zum schnelleren Zugriff auf Disketten können beispielweise einige Cache-Puffer hinzugefügt werden. Fügen Sie Verzeichnisse mit häufig verwendeten Befehlen dem Suchpfad hinzu. Außerdem kann angegeben werden, daß das Programm "Blanker" gestartet werden soll. Für die in diesem Beispiel genannten Punkte können Sie folgendes eingeben:

```
ADDBUFFERS >NIL: DF0: 25
PATH >NIL: SYS:MeinBefehl ADD
RUN Blanker CX_POPUP=NO SECONDS=600 ANIMATION=YES
```

Anschließend speichern Sie die Datei und verlassen das Programm durch Drücken der Tasten Esc,X,Eingabetaste. Beim nächsten Systemstart bzw. Neustart werden die genannten Befehle ausgeführt. Sollen weitere Befehle hinzugefügt werden, geben Sie erneut **RUN ED S:User-startup** ein, um die Datei zu edieren.

8.1.9 Erstellen einer Zuweisung

Bei Diskettensystemen bedeutet ein wie in Abb. 8-1 dargestelltes Dialogfenster, daß Sie die angegebene Diskette einlegen müssen.



Abb. 8-1. Systemdialogfenster (Beispiel)

Der häufigste Grund für die Anzeige eines solchen Dialogfensters bei Festplattensystemen ist mit Ausnahme eines falsch positionierten Doppelpunkts (:) das Fehlen einer Zuweisung (Befehl ASSIGN) vor einer gerade aufgerufenen Anwendung. Die Zuweisungen erfolgen oftmals durch ein Installationsprogramm, aber nicht alle Anwendungen verfügen über ein Installationsprogramm oder nur über ein Installationsprogramm, das nicht auf allen Systemen ausgeführt werden kann.

Über die Zuweisung wird der Anwendung mitgeteilt, daß die erforderlichen Daten nicht auf Diskette, sondern auf der Festplatte gespeichert sind. Wenn Sie die Anwendung regelmäßig verwenden, fügen Sie eine entsprechende ASSIGN-Anweisung in Ihrer Datei "User-startup" hinzu. Wenn das Dialogfenster zum ersten Mal angezeigt wird, können Sie jedoch die Zuweisung über die Shell eingeben. Anschließend wählen Sie im Dialogfenster "Wiederholen" aus.

Zum Fortsetzen Ihrer Arbeit mit einem Programm, das von der Disk "ProFusion" auf den Datenträger "Work:" kopiert wurde, geben Sie folgende Anweisung ein:

```
1> ASSIGN ProFusion: Work:ProFusion
```

Dabei gibt das erste Argument den Namen des erforderlichen Datenträgers an. Das zweite Argument enthält den Namen des Geräts und des Verzeichnisses, auf dem bzw. in dem "ProFusion" installiert wurde. Nach Eingabe des Befehls klicken Sie im

Dialogfenster "Wiederholen" an. Falls die Anweisung fehlerfrei ist, wird die Anwendung ordnungsgemäß ausgeführt. Fügen Sie anschließend die eingegebene Anweisung in der Datei "User-startup" hinzu. Schlägt die Anweisung fehl, müssen Sie das zweite Argument ändern.

8.1.10 Zugreifen auf die erweiterten Menüs von ED

Über die vorgegebene Datei "S:Ed-startup" wird eine vereinfachte Gruppe von Menüs für den Texteditor ED konfiguriert. Darüber hinaus ist eine erweiterte Gruppe von Menüs in den Texteditor ED integriert, die weitere Optionen enthält. Diese erweiterte Gruppe wird bereitgestellt, wenn Sie die Datei "Ed-startup" so umbenennen, daß sie nicht ausgeführt wird.

Soll auf die erweiterten Menüs des Editors ED zugegriffen werden, geben Sie folgenden Befehl ein:

```
1> RENAME S:Ed-startup TO S:Ed-startup.nein
```

8.1.11 Arbeiten mit einer einzigen Shell

Sie können zwar gleichzeitig mit mehreren Shell-Fenstern arbeiten, aber möglicherweise wollen Sie vermeiden, daß sich auf dem Workbench-Bildschirm mehrere Shell-Fenster überlagern. Wenn Sie ein Programm von einer Shell aus starten, wird das zugehörige Shell-Fenster in der Regel während der Ausführung des Programms vom jeweiligen Programm übernommen. Zwangsläufig müssen Sie zur Eingabe weiterer Befehle eine andere Shell öffnen. Mit den folgenden Methoden können Sie dies vermeiden und eine beliebige Anzahl von Programmen von einem Shell-Fenster aus starten.

Soll ein Programm so im Hintergrund ausgeführt werden, daß die Shell-Eingabeaufforderung wieder angezeigt wird, verwenden Sie den Befehl RUN:

```
1> RUN DMC OPT def RHYME on
[CLI 2]
1>
```

Die Meldung in eckigen Klammern gibt die Nummer des neuen Prozesses an, der zum Ausführen des Programms gestartet wurde. Die Eingabeaufforderung wird sofort wieder angezeigt.

Selbst wenn ein Programm mit dieser Methode gestartet wurde, kann das Shell-Fenster erst verlassen werden, wenn alle von der Shell aus gestarteten Programme verlassen wurden. Um dies zu vermeiden, können Sie ein Programm unter Verwendung der Umleitungsfunktionen von der Shell lösen.

Zum Lösen eines Programms von der Shell rufen Sie es mit dem Befehl RUN auf. Dabei geben Sie an, daß die Ausgabe zum Gerät NIL: umgeleitet werden soll:

```
1> RUN >NIL: MultiView 8SVX/Muster
[CLI 2]
1>
```

Jetzt können Sie das Shell-Fenster ggf. schließen.

Hinweis Durch Umleitung zum Gerät NIL: wird auch vermieden, daß die vom Programm für die Konsole generierten Ausgabedaten im Shell-Fenster erscheinen.

8.1.12 Hinzufügen von Piktogrammen

Soll ein Piktogramm für eine Datei oder ein Verzeichnis hinzugefügt werden, können Sie unter Verwendung des Programms "IconEdit" ein Piktogramm erstellen. Häufig ist es jedoch einfacher, ein bereits vorhandenes Piktogramm mit einem Shell-Befehl zu kopieren.

Die kopierte .info-Datei muß der erforderlichen Piktogrammart entsprechen. (Nachträglich kann diese auch mit IconEdit angepaßt werden.) Beim Kopiervorgang geben Sie dem Piktogramm einen Namen, der der Datei bzw. dem Verzeichnis entspricht, und fügen die Erweiterung ".info" hinzu. Je nach Datei ändern Sie gegebenenfalls über den Menüpunkt "Informationen..." das für ein Projektpiktogramm angegebene Standardprogramm oder die Merkmale des kopierten Programmpiktogramms.

Hinweis Der Name, der unterhalb des Piktogramms angezeigt wird, entspricht (einschließlich Groß- und Kleinschreibung) dem im Befehl COPY für die .info-Datei angegebenen Namen. Die Groß- und Kleinschreibung ist unabhängig von der zugehörigen Datei bzw. dem zugehörigen Verzeichnis.

Soll eine Datei PCX im Verzeichnis "DataTypes" mit einem Piktogramm versehen werden, kopieren Sie die .info-Datei eines bereits im Verzeichnis vorhandenen Dateientyps:

```
1> COPY DataTypes/ILBM.info TO DataTypes/PCX.info
```

Beim Öffnen des Fensters "DataTypes" bzw. bei Auswahl von "aktualisieren" im Menü "Fenster" wird das Piktogramm PCX angezeigt.

Hinweis Wenn Sie das zu kopierende Piktogramm fixiert haben, bleibt das neue Piktogramm an der Position des ursprünglichen Piktogramms und überlagert es. Ziehen Sie das neue Piktogramm an eine andere Position. Anschließend fixieren Sie es, damit die beiden Piktogramme voneinander getrennt werden.

Soll Disk "VidTools" mit einem benutzerspezifischen Piktogramm versehen werden, kopieren Sie die gewünschte .info-Datei (vom Disk-Typ) in das Hauptverzeichnis der Diskette. Dabei geben Sie dem Piktogramm den Namen "disk.info":

```
1> COPY SYS:disk.info TO VidTools:disk.info
```

Sie müssen die Disk entnehmen und erneut einlegen oder das System erneut starten, damit das neue Disk-Piktogramm auf der Workbench angezeigt wird.

8.1.13 Eds - Benutzerfreundliches Erstellen von Befehlsdateien

Zum leichteren Erstellen und Edieren von Befehlsdateien gehen Sie wie folgt vor:

Erstellen Sie eine Befehlsdatei mit den folgenden Zeilen und speichern Sie sie als "S:Eds". (Ein Beispiel zur Erstellung einer Befehlsdatei finden Sie in Kapitel 8.1.8 "*Erstellen einer Datei "User-startup*" auf Seite 8-10.)

```
.KEY script/A
ED S:<script>
FAILAT 11
IF EXISTS S:<script>
   PROTECT S:<script> srwd
ENDIF
```

Geben Sie **PROTECT S:Eds srwd** ein, um das Schutzbit "s" für die Befehlsdatei zu aktivieren. Jetzt können Sie unter Verwendung von "Eds" Befehlsdateien erstellen und edieren, ohne jedes Mal deren Position festlegen und das Schutzbit "s" aktivieren zu müssen. Geben Sie lediglich **Eds** gefolgt vom Namen der gewünschten Befehlsdatei ein.

Wenn Sie die Datei speichern und ED verlassen, wird die bearbeitete Befehlsdatei im Verzeichnis "S:" unter dem angegebenen Namen gespeichert. Das Schutzbit "s" wird automatisch aktiviert, so daß Sie die neue Befehlsdatei ohne den Befehl EXECUTE von jeder Shell aus aufrufen können.

8.2 Gelegentliche Aufgaben

Die in diesem Abschnitt beschriebenen Aufgaben sind zwar seltener erforderlich, aber fast jeder Benutzer muß die genannten Arbeitsschritte einmal ausführen. Bei diesen Beispielen wird davon ausgegangen, daß Sie einigermaßen mit AmigaDOS und der Shell vertraut sind.

8.2.1 Erstellen von Alias-Namen zur Reduzierung von Tastenanschlägen

Die weiter unten als Vorschlag genannten Alias-Namen dienen zur Beschleunigung Ihrer Arbeit innerhalb der Shell, da durch sie die Anzahl der Tastenanschläge reduziert wird, die für allgemeine Befehle erforderlich sind.

Sollen diese Alias-Namen als globale Alias-Namen aktiviert werden, edieren Sie die Datei "S:Shell-startup" (ähnlich wie oben im Kap. 8.1.8) und fügen Sie folgende Zeilen hinzu:

```
ALIAS c0 CD DF0:
ALIAS cs CD SYS:
ALIAS css CD S:
ALIAS d0 DIR DF0:
ALIAS dr DIR RAM:
ALIAS qdir DIR ~(#?.info)
ALIAS ls LIST
ALIAS cp COPY
ALIAS cc COPY [] CLONE
ALIAS del DELETE
ALIAS ren RENAME
ALIAS ns NEWSHELL
ALIAS es ENDSHELL
ALIAS pf printfiles
ALIAS fmt0 FORMAT DRIVE DF0: NAME [] FFS NOICONS
 DIRCACHE
ALIAS formq FORMAT DRIVE DF0: NAME [] NOICONS
    QUICK
ALIAS edus RUN ED S:User-startup
ALIAS edsh RUN ED S:Shell-startup
ALIAS ednew RUN ED RAM:newfile
ALIAS chip ECHO "Es sind `avail chip` Bytes im
 Chip-RAM frei."
```

Ändern Sie die Alias-Namen Ihren Erfordernissen entsprechend. Verwenden Sie die aufgelisteten Alias-Namen als Modelle für Ihre eigenen Alias-Namen.

8.2.2 Anpassen des Befehls NEWSHELL

Das Aussehen und die Funktionsweise eines Shell-Fensters können Sie über das Argument WINDOW des Befehls NEWSHELL steuern. Das Argument ermöglicht Ihnen die Angabe spezifischer Größen, Positionen und Funktionen für das Shell-Fenster. Im folgenden werden zwei Beispiele für Fensterspezifikationen beschrieben.

Zum Öffnen eines praktischen Shell-Fensters auf Ihrem Workbench-Bildschirm geben Sie folgenden Befehl über die Shell oder als Befehlszeile in der Datei "User-startup" ein:

```
NEWSHELL CON:100/100/400/100/AShell/CLOSE/
 ALT0/12/640/388
```

Daraufhin wird ein kleines Shell-Fenster erstellt. Dieses Fenster trägt den Titel "AShell" und belegt nur die rechte Seite des Workbench-Fensters, so daß die Disk-Piktogramme nicht verdeckt werden. Dieses Fenster verfügt über ein Schließsymbol, und auf hochauflösenden Bildschirmen mit Zeilensprungverfahren (engl. High-Res Interlace) füllt es den gesamten Bildschirm mit Ausnahme der Titelzeile aus, wenn Sie das Zoom-Symbol auswählen.

Zum Öffnen eines Shell-Fensters auf einem Public-Schirm, z. B. dem Terminal-Bildschirm eines Telekommunikationsprogramms, verwenden Sie eine Fensterspezifikation mit der Option SCREEN:

```
CON:0/20///???/CLOSE/SCREENTerm
```

Daraufhin wird ein Fenster erstellt, das mit drei Fragezeichen (???) betitelt ist. Dieses Fenster ist so breit wie möglich und hat die niedrigstmögliche Höhe. Das Fenster wird auf dem Public-Schirm "Term" geöffnet, falls dieser vorhanden ist. Ansonsten wird das Shell-Fenster auf dem Workbench-Bildschirm geöffnet.

8.2.3 Ändern der Eingabeaufforderung

Die Shell-Eingabeaufforderung kann mit dem Befehl PROMPT problemlos geändert werden. In der Eingabeaufforderung kann jeder angegebene Text hinzugefügt werden. Sie können Platzhalter, die zur Anzeige von Prozeßnummer, aktuellem Verzeichnis und Rückgabecode dienen, einfügen, neu anordnen oder auslassen.

Außerdem können Sie Escape-Sequenzen in die Eingabeaufforderung integrieren, die zur farblichen Hervorhebung der Eingabeaufforderung gegenüber der Befehlszeile und den Ausgabedaten des Befehls dienen können. Darüber hinaus können Befehle in die Eingabeaufforderung eingefügt werden, indem Sie sie in umgekehrte Apostrophe setzen.

In Abb. 8-2 sind zwei Methoden zum Ändern der Eingabeaufforderung dargestellt. Bei der ersten Methode werden Escape-Sequenzen verwendet, so daß die Eingabeaufforderung in Fettschrift mit Farbe 2 (weiß) angezeigt wird. Anhang D enthält eine Liste der verfügbaren Escape-Sequenzen. Bei der zweiten Methode wird die übliche Position der Ersetzungsoperatoren geändert. Außerdem wird der Befehl DATE mit umgekehrten Apostrophen in die Eingabeaufforderung integriert. Das abschließende Zeichen wird in ein Dollarzeichen ($) geändert.

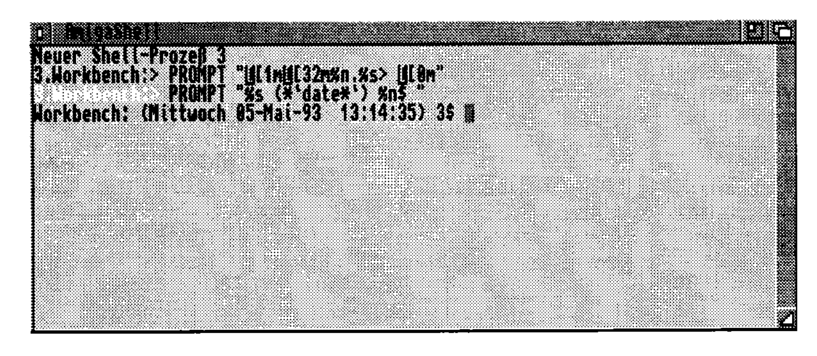


Abb. 8-2. Verwendung des Befehls PROMPT (Beispiele)

8.2.4 Erstellen eines benutzerspezifischen RAM-Disk-Piktogramms

Damit ein benutzerspezifisches Piktogramm für die RAM-Disk angezeigt wird, müssen Sie eine COPY-Anweisung in der Datei "User-startup" hinzufügen. Zunächst erstellen Sie das Piktogramm mit dem Programm "IconEdit". Beachten Sie, daß es sich um ein Piktogramm für eine Disk handeln muß. Anschließend speichern Sie das Piktogramm auf Ihrer Start-Disk als "DEVS:Ramdisk.info".

Hinweis In diesem Beispiel wird zwar das Verzeichnis DEVS: verwendet, aber das Piktogramm kann in einem beliebigen Verzeichnis gespeichert werden. Die Datei wird selbst im Modus "Inhalt anzeigen alle Dateien" nicht angezeigt, da Disk-Piktogramme nur auf der Workbench sichtbar sind.

Soll Ihr benutzerspezifisches RAM-Disk-Piktogramm auf der Workbench angezeigt werden, fügen Sie die folgende Zeile in der Datei "S:User-startup" hinzu. Anschließend speichern Sie die geänderte Datei und starten Sie den Computer erneut:

```
COPY DEVS:Ramdisk.info TO RAM:disk.info
```

Soll die RAM-Disk auch umbenannt werden, fügen Sie eine RELABEL-Anweisung hinter der COPY-Anweisung in der Datei "User-startup" hinzu. Beispiel:

```
RELABEL RAM: ramname
```

8.2.5 Löschen von Dateien mit Piktogrammen

Zum Löschen eines Piktogramms und der .info-Datei der zugehörigen Datei mit einem einzigen Befehl gehen Sie wie folgt vor:

1. Erstellen Sie mit Eds (s. o.) die folgende Befehlsdatei und speichern Sie sie als "S:Delinf":

```
.KEY Datei/A
DELETE <Datei> <Datei>.info QUIET
```

2. Geben Sie **Delinf** gefolgt vom Namen der Datei ein.

8.2.6 Testen von Befehlen

Bisweilen müssen Sie die Ergebnisse bestimmter Befehle überprüfen, bevor Sie sie tatsächlich für vorhandene Dateien verwenden können. Dies gilt insbesondere, wenn Sie komplexe Namensmuster mit möglicherweise zerstörenden Befehlen (z. B. DELETE), oder Escape-Sequenzen verwenden, bei denen die Ergebnisse nur schwer

vorhersagbar sind. Es gibt verschiedene schnelle und sichere Methoden zum Testen von Befehlen.

Zum Testen eines möglicherweise zerstörenden Befehls mit Namensmuster gehen Sie wie folgt vor:

1. Geben Sie einen nicht zerstörenden Befehl (z. B. LIST) mit dem beabsichtigten Namensmuster ein. Beispiel:

   ```
   LIST ~(#?.info|#?.c|[0-9]#?)
   ```

2. Überprüfen Sie anhand der Ausgabedaten, ob das Namensmuster die gewünschten Dateien anspricht.
3. Ändern Sie ggf. das Namensmuster und wiederholen Sie den Befehl, bis die gewünschten Dateien aufgelistet werden.
4. Geben Sie anschließend den geplanten Befehl mit demselben Namensmuster ein.

Zum Überprüfen der Ergebnisse von Escape-Sequenzen gehen Sie wie folgt vor:

1. Geben Sie den Befehl ECHO mit den beabsichtigten Escape-Sequenzen und ein Beispielwort ein. Beispiel:

   ```
   ECHO "*E[1mBOLD*E[0m"
   ```

2. Überprüfen Sie, ob die Ausgabe Ihren Wünschen entsprechend erfolgt.
3. Ändern Sie ggf. die Escape-Sequenzen und wiederholen Sie den Befehl, bis das Ergebnis Ihren Wünschen entspricht.
4. Verwenden Sie anschließend dieselben Escape-Sequenzen im geplanten Befehl (z. B. PROMPT).

Zum Löschen des Fensterinhalts und Zurücksetzen der Modi aller Escape-Sequenzen auf die vorgegebenen Werte geben Sie folgendes ein:

```
Esc,c,Eingabetaste
```

Sie müssen den Kleinbuchstaben "c" verwenden. Nach Löschen des Fensterinhalts und Zurücksetzen der Modi wird in der obersten Zeile des Shell-Fensters die ansonsten folgenlose Fehlermeldung **`:Unbekannter Befehl`** angezeigt.

Zum Erstellen einer Testdatei in der RAM-Disk gehen Sie wie folgt vor:

```
1> ECHO "Dies ist nur ein Test." TO RAM:foo
```

Mit diesem Befehl wird ein kleine erweiterbare Datei erstellt, die zum Testen anderer Befehle dient. Es empfiehlt sich, derartige kleine Dateien im RAM: (Arbeitsspeicher) zu erstellen, um zu vermeiden, daß Ihre Disks mit kleinen nutzlosen Dateien überfüllt werden. Traditionell nennt man solche Dateien "foo" oder "bar" oder "test".

Zum sicheren Testen von Befehlen für wichtige Dateien kopieren Sie die Testdateien in ein Verzeichnis in der RAM-Disk und testen Sie die Befehle mit diesen Dateien:

```
1> COPY Arbeit:MeineDateien/#? TO RAM:Testdir
```

8.2.7 Erstellen einer Befehlsdatei zum Verlegen von Dateien

Bei AmigaDOS gibt es keinen Befehl MOVE zum Verlegen von Dateien oder Verzeichnissen. In der Regel stehen bei AmigaDOS zwei Methoden zum Verlegen von Dateien zur Verfügung: der Befehl RENAME oder eine Kombination der Befehle COPY und DELETE.

Mit dem Befehl RENAME können Sie eine Datei nur dann verlegen, wenn die Zieldatei sich auf demselben Datenträger befindet. Bei dem Versuch, eine Datei mit dem Befehl RENAME auf ein anderes Gerät zu verlegen, wird eine Fehlermeldung angezeigt. Die Methode mit Kopieren und Löschen der Dateien ist in jeder Situation möglich, erfordert aber mehr Aufwand.

Zum Erstellen eines Befehls MOVE (Verlegen), mit dem Sie eine Datei mit einem einzigen Befehl innerhalb von Geräten oder von einem Gerät auf ein anderes verlegen können, erstellen Sie eine Befehlsdatei mit Eds (s. o.):

```
1> Eds MOVE
```

Geben Sie folgende Befehle im ED-Fenster ein:

```
.KEY Quelle/A,Ziel/A
.BRA {
.KET }
FAILAT 21
RENAME >NIL: {Quelle} TO {Ziel}
IF WARN
   COPY {Quelle} TO {Ziel}
   IF WARN
      ECHO "Die Datei {Quelle} konnte nicht verlegt
werden."
      QUIT 20
   ENDIF
   DELETE {Quelle} QUIET
ENDIF
ECHO {Quelle} "wurde verlegt nach" {Ziel}
```

Speichern Sie die Befehlsdatei und verlassen Sie ED. Nach dem Verlassen wird das Schutzbit "s" automatisch aktiviert.

Diese Befehlsdatei können Sie wie einen Befehl verwenden. Geben Sie den Befehl MOVE gefolgt von zwei Argumenten ein: dem aktuellen Pfad der zu verlegenden Datei und den Pfad zum gewünschten Standort.

8.2.8 Löschen mit interaktivem Befehl DIR

Der Befehl DIR verfügt über einen interaktiven Modus, bei dem nach jeder angezeigten Datei bzw. jedem angezeigten Verzeichnis eine Pause gemacht wird, die Ihnen die Eingabe eines Befehls aus einer Auswahl von mehreren einfachen Befehlen ermöglicht. Sie haben unter anderem die Möglichkeit, das Objekt zu löschen. Dies ist in einigen Situationen hilfreich.

Wenn Sie beispielsweise einer Datei versehentlich den Namen #? (die Jokerzeichenkombination für alle denkbaren Namen) geben, können Sie diese Datei nicht mit den üblichen Methoden löschen. Wenn Sie den Befehl **DELETE #?** eingeben, werden nämlich alle Dateien und Unterverzeichnisse im Verzeichnis gelöscht, einschließlich wichtiger Dateien.

Mit dem interaktiven DIR-Befehl können Sie dieses Problem lösen. Außerdem kann der Befehl als allgemeines Abfrage- und Löschprogramm verwendet werden. Dies ist hilfreich, wenn in einem

umfangreichen Verzeichnis viele Dateien gelöscht werden sollen, aber die Dateinamen so unterschiedlich sind, daß sich für den Befehl DELETE keine Namensmuster anbieten.

Zum sicheren Löschen verschiedener Dateien auf einer Disk "Schrott" mit dem interaktiven DIR-Befehl geben Sie folgendes ein:

```
1> DIR Schrott: INTER
```

Vom Befehl DIR werden die Namen der Dateien auf der Disk "Schrott" nacheinander in alphabetischer Reihenfolge aufgelistet. Hinter jedem Namen wird eine Eingabeaufforderung in Form eines Fragezeichens angezeigt. Drücken Sie die Eingabetaste, um zur nächsten Datei zu wechseln, oder die Taste E, um in ein angezeigtes Verzeichnis zu wechseln. Wenn Sie eine nicht mehr gewünschte Datei sehen, geben Sie zum Löschen der Datei die Buchstabenfolge **DEL** ein. Bei Eingabe von **Q** wird der interaktive Modus des Befehls DIR verlassen.

8.2.9 Generieren von Befehlsdateien mit LIST LFORMAT

Einer der Hauptverwendungszwecke der Option LFORMAT des Befehls LIST ist die automatische Erstellung von Befehlsdateien, die zur gleichzeitigen Verarbeitung mehrerer Dateien dienen. Mit einer LIST-Anweisung mit der Option LFORMAT können Sie eine einfache Befehlsdatei erstellen. Dazu können Sie das Argument TO zum Umleiten der LIST-Ausgabe in eine Datei sowie Namensmuster verwenden. Anschließend können Sie die Befehlsdatei mit einem Texteditor aufrufen und sie ggf. Ihren Erfordernissen entsprechend anpassen, bevor Sie sie starten.

Soll eine Befehlsdatei erstellt werden, mit deren Hilfe alle Dateien im aktuellen Verzeichnis durch Anhängen der Erweiterung .IFF an die bisherigen Dateinamen umbenannt werden, und sollen diese Dateien in das Verzeichnis "BildDat" auf der Partition "Work" verlegt werden, geben Sie folgenden Befehl ein:

```
1> LIST #? TO T:renamer LFORMAT="RENAME %P%N TO
Work:BildDat/%N.IFF"
```

Anschließend geben Sie **ED T:renamer** ein, um die Befehlsdatei zu überprüfen. Diese sollte dem folgenden Beispiel entsprechen. Wenn aufgrund des Namensmusters nicht zu verarbeitende Dateien aufgelistet werden oder die Zeichenkette bei LFORMAT nicht dem erwarteten Ergebnis entspricht, können Sie entweder die Befehlsdatei edieren oder den Befehl ändern und erneut eingeben.

```
RENAME Bilder:Peter TO Work:BildDat/Peter.IFF
RENAME Bilder:Sven TO Work:BildDat/Swen.IFF
RENAME Bilder:Paul TO Work:BildDat/Paul.IFF
RENAME Bilder:Maria TO Work:BildDat/Maria.IFF
RENAME Bilder:Anne TO Work:BildDat/Anne.IFF
```

Wenn die Befehlsdatei Ihren Erfordernissen entspricht, verlassen Sie den Texteditor ED. Anschließend geben Sie den Befehl **EXECUTE T:renamer** ein.

Kapitel 8.3.5 "*Rekursive AmigaDOS-Befehlsdateien*" auf Seite 8-32 enthält ein komplexeres Beispiel zur Verwendung des Befehls LIST LFORMAT.

8.2.10 Anpassen der LIST-Ausgabe

Außerdem können Sie die Option LFORMAT verwenden, um das Ausgabeformat des Befehls LIST für spezielle Zwecke anzupassen. Weisen Sie der Befehlszeile einen Alias-Namen zu, um sich deren spätere Verwendung zu erleichtern.

Soll ein Alias-Name für einen LIST-Befehl erstellt werden, mit dem nur die Informationen über Dateien angezeigt werden, die nach dem angegebenen Datum erstellt bzw. geändert wurden, und sollen das Datum an erster Stelle und die Schutzbits sowie die Uhrzeit nicht angezeigt werden, geben Sie folgende Befehlszeile ein:

```
1> ALIAS lsince LIST FILES SINCE [] LFORMAT="%D
%-25N %L"
```

Ist "S:" das aktuelle Verzeichnis, sind die Ausgabedaten des Befehls **lsince 01-sep-92** mit dem folgenden Beispiel vergleichbar:

```
19-Oct-92   PCD                715
30-Nov-92   Startup-sequence  1360
Freitag     Shell-startup      671
Gestern     User-startup       609
```

8.2.11 Ausführen von Befehlsdateien mit Hilfe von ICONX

Wenn Sie es bevorzugen, möglichst mit der Maus zu arbeiten, oder den Amiga für einen Benutzer konfigurieren, der ausschließlich mit der Workbench arbeitet, erweist sich das Programm "ICONX" als hilfreich. Wenn Sie C:ICONX als Standardprogramm für ein Projektpiktogramm verwenden, können Sie eine Befehlsdatei (oder einen Befehl, für den keine Workbench-Entsprechung vorhanden ist) durch Anklicken des zugehörigen Piktogramms starten. Dabei wird die Befehlsdatei so gestartet, als ob Sie das Piktogramm für ein vorhandenes Workbench-Programm öffnen.

Diese Form des Programmstarts bietet aber auch für den erfahrenen Benutzer Vorteile. So werden z. B. bei der Verwendung von Befehlsdateien die vor dem Programmstart auszuführenden Schritte (z. B. Laden bestimmter Einstellungen der Voreinsteller-Editoren) erleichtert. Außerdem kann die Priorität des Programms problemlos geändert werden. Diese Methoden werden am folgenden Beispiel erläutert.

Soll eine Anwendung "AltApp" über ein ICONX-Piktogramm gestartet werden und sollen dabei zunächst spezielle Einstellungen des Zeichensatz-Voreinstellers (Font) geladen und die Task-Priorität der Anwendung geändert werden, erstellen Sie folgende Befehlsdatei für das Piktogramm:

```
CD SYS:
Prefs/Font FROM SYS:Prefs/Presets/defscrn.pre USE
CHANGETASKPRI -1
AltApp
```

8.2.12 Vermeiden der Datenausgabe bei Befehlsdateien

Bei der Verarbeitung von Befehlsdateien sollen häufig Befehle ausgeführt werden, ohne daß deren übliche Ausgabedaten im Shell-Fenster angezeigt werden. Dadurch kann vermieden werden, daß eine überflüssige Folge von Meldungen angezeigt bzw. ein Ausgabefenster zu einem ungünstigen Zeitpunkt geöffnet wird.

Soll vermieden werden, daß Daten an die Konsole ausgegeben werden, leiten Sie die Ausgabedaten des Befehls wie folgt mit dem Argument `>NIL:` an ein Pseudogerät um:

```
1> DELETE >NIL: T:TempDat
```

Durch das Argument >NIL: wird vermieden, daß die Meldung **`T:TempDat Deleted`** (T:TempDat gelöscht) bzw. Fehlermeldungen auf dem Bildschirm angezeigt werden.

Soll die Konsolenausgabe mit Ausnahme von Fehlermeldungen vermieden werden, verwenden Sie bei Befehlen, bei denen diese Option verfügbar ist, die Option QUIET:

```
1> DELETE T:TempDat QUIET
```

Die Meldung **`T:Tempfile Deleted`** wird nicht angezeigt. Ist die Datei "T:TempDat" jedoch nicht vorhanden, erscheint eine diesbezügliche Fehlermeldung.

8.2.13 Eingeben und Testen von ARexx-Makros

Zum leichteren Eingeben und Testen von ARexx-Makros gehen Sie wie folgt vor:

Erstellen Sie mit Eds (s. o.) die folgende AmigaDOS-Befehlsdatei und speichern Sie sie unter dem Namen "S:Edrx":

```
.KEY mac
ED REXX:<mac>.rexx
PROTECT S:<mac>.rexx +S-E
RX REXX:<mac>
```

Verwenden Sie diese Befehlsdatei beim Ausprobieren von ARexx-Makros. Geben Sie **`Edrx`** gefolgt vom Namen des jeweiligen Makros ein. Über "Edrx" wird der Editor aufgerufen, ohne daß Sie die Erweiterung ".rexx" eingeben müssen. Beim Verlassen des Editors wird das ARexx-Interpretationsprogamm (engl. Interpreter) RX direkt mit Ihrem Makro als Argument aufgerufen.

8.2.14 Sortieren und Verknüpfen von Dateien

Wenn Sie beispielsweise regelmäßig per DFÜ eine neue Liste von Dateien laden, sollen diese Dateien möglicherweise einer Liste

vorhandener Dateien hinzugefügt und alle Einträge nach Datum sortiert werden. Jeder Dateiname habe dasselbe Präfix und ein Suffix mit dem zugehörigen Datum, z. B. NeuDat.941009 für die Dateiliste vom 9. Oktober 1994.

Soll eine einzige Dateiliste erstellt werden, in der die Einträge nach Datum sortiert sind, verwenden Sie die Befehle JOIN und SORT mit Namensmuster:

```
JOIN NeuDat.#? TO RAM:Temp.joined
SORT RAM:Temp.joined TO NeuDat.sortiert
DELETE RAM:Temp.joined
```

8.3 Komplexere Aufgaben

In den folgenden Beispielen werden anspruchsvollere Aufgaben für Benutzer erläutert, die sich ausgezeichnet mit AmigaDOS auskennen.

8.3.1 Überprüfen von Softwareversionen

Bisweilen müssen Sie die Abläufe in einer Befehlsdatei in Abhängigkeit von der Softwareversion ändern, mit der der Benutzer arbeitet. Versionsnummern können in einer Befehlsdatei leicht überprüft werden.

Soll überprüft werden, ob eine Befehlsdatei auf einem Amiga mit Systemsoftware der Version 2 ausgeführt wird, fügen Sie z. B. vor dem versionsabhängigen Teil der Befehlsdatei folgende Zeilen ein:

```
VERSION >NIL: 37
IF WARN
    ECHO "Zeit für eine Systemaktualisierung."
    QUIT
ELSE
    ECHO "Version 2 oder höher.       Gut!    "
    ECHO "Weiter geht's."
ENDIF
```

Versionsnummern unter 37 stehen für Versionen vor Version 2; Versionsnummer 38 steht für Version 2.1, 39 für Version 3 und 40 für Version 3.1.

8.3.2 Auslagern nicht benutzter Zeichensätze und Bibliotheken

Wenn Zeichensätze (Fonts) und Bibliotheken (Libraries) in den Speicher geladen werden, verbleiben sie selbst dann im Speicher, wenn sie zur Zeit nicht verwendet werden. Sie werden nur dann automatisch aus dem Speicher entfernt, wenn der durch sie belegte Speicherplatz für einen anderen Zweck benötigt wird. In einigen Fällen empfiehlt es sich jedoch, daß Sie nicht benutzte Ressourcen selber aus dem Speicher entfernen.

Sie wollen beispielsweise einen Zeichensatz edieren und anschließend testen oder die Versionsnummer der Bibliothek auf einer neuen Disk abfragen, aber der Amiga kann den neuen Zeichensatz bzw. die neue Bibliothek scheinbar nicht finden. Dies liegt daran, daß von AmigaDOS möglichst immer der Zeichensatz bzw. die Bibliothek verwendet wird, der/die sich im Speicher befindet. Über die Option FLUSH des Befehls AVAIL können Sie nicht benutzte Ressourcen aus dem Speicher entfernen und die Fragmentierung des Speichers verringern.

Soll ein nicht benutzter Zeichensatz bzw. eine nicht benutzte Bibliothek aus dem Speicher ausgelagert werden, ohne den Computer erneut zu starten, geben Sie folgendes ein:

```
1> AVAIL FLUSH
```

Damit ein Zeichensatz bzw. eine Bibliothek aus dem Speicher entfernt werden kann, darf er/sie zur Zeit nicht von der Workbench oder einer anderen Anwendung verwendet werden.

8.3.3 Erstellen von AmigaDOS-Schleifen mit EVAL

Soll in einem Skript eine Schleife programmiert werden, vor der Sie ggf. aufgefordert werden, die Anzahl der Durchläufe anzugeben, gehen Sie wie folgt vor:

Geben Sie mit Eds (s. o.) folgende Befehlsdatei ein und speichern Sie sie als "MSchleife":

```
.KEY Schleife
; Klammern bei Ersetzungsangaben ändern, da
; Befehlsdatei < und > zur Umleitung verwendet:
.BRA {
.KET }
; Überprüfen, ob Benutzer Argument für Anzahl
; Schleifendurchläufe angegeben hat. Falls nicht:
IF NOT {Schleife}
    ECHO "Bitte Anzahl der Durchläufe angeben"
    ECHO "und die Eingabetaste drücken: " NOLINE
    SETENV >NIL: Schleife{$$} ?
ELSE
    ; Argument vorhanden, Wert speichern
    ECHO >ENV:Schleife{$$} {Schleife}
ENDIF
;
LAB start                ; Schleifenanfang
ECHO "Schleifenanzahl " NOLINE ; hier Anzahl der
TYPE ENV:Schleife{$$}    ; Befehlswiederholungen
EVAL <ENV:Schleife{$$} >NIL: TO=T:Qwe{$$} VALUE2=1
    OP=- ?
    ; Alternativ: ECHO >T:Qwe{$$} `EVAL
    $Schleife{$$} - 1`
TYPE >ENV:Schleife{$$} T:Qwe{$$}
IF VAL $Schleife{$$} GT 0
    SKIP start BACK      ;Schleife noch aktiv
ENDIF
;
DELETE ENV:Schleife{$$} T:Qwe{$$} QUIET ;aufräumen
ECHO "Schleife beendet"
```

Wenn Sie diese Befehlsdatei starten, ohne ein Argument anzugeben, werden Sie aufgefordert, die Anzahl der gewünschten Durchläufe anzugeben. Der angegebene Wert wird als Anfangswert für die Schleife verwendet. Geben Sie über folgenden Befehl einen Wert an:

```
1> MSchleife 5
```

Die folgenden Ausgaben werden angezeigt:

```
Schleifenanzahl 5
Schleifenanzahl 4
Schleifenanzahl 3
Schleifenanzahl 2
Schleifenanzahl 1
Schleife beendet
```

Bei dieser Schleife besteht die einzige Aktion darin, die Anzahl der verbleibenden Schleifendurchläufe anzuzeigen. Unter Verwendung der Umgebungsvariablen "Schleife{$$}" können Sie jedoch komplexere Aktionen angeben.

Beim ersten IF-Block wird überprüft, ob beim Aufrufen der Befehlsdatei ein Argument angegeben wurde. Ist dies nicht der Fall, wird der Benutzer zur Eingabe eines Werts aufgefordert. In beiden Fällen wird der Wert in der Variablendatei "ENV:Schleife{$$}" gespeichert. Durch den Operator {$$} wird die aktuelle Prozeßnummer an den Namen "Schleife" angehängt, um einen eindeutigen Dateinamen zu erhalten. Dadurch sollen mögliche Konflikte beim Multitasking vermieden werden. Der Dateiname lautet beispielsweise "ENV:Schleife4".

Innerhalb der Schleife wird durch die Verknüpfung des Befehls ECHO mit dem Befehl TYPE das Wort "Schleifenanzahl" gefolgt von der als Argument angegebenen Nummer angezeigt. Beim ersten Durchlauf erscheint **`Schleifenanzahl 5`**.

Der Befehl EVAL verwendet die Nummer in der Datei "ENV:Schleife{$$}" als <Wert1> (VALUE1). Daher muß am Ende der Zeile ein Fragezeichen stehen. <Wert2> (VALUE2) ist 1 und die auszuführende Operation Subtraktion. Die Ausgabedaten des Befehls EVAL werden in die Datei "T:Qwe{$$}" geschrieben. Mit dem nächsten Befehl TYPE wird der Wert aus der Datei "T:Qwe($$)" an die Datei "ENV:Schleife{$$}" kopiert. Durch diese beiden Zeilen wird jeweils 1 vom Wert in der Datei "ENV:Schleife{$$}" subtrahiert. Die EVAL-Zeile kann alternativ und eleganter mit umgekehrten Hochkommas geschrieben werden.

Über die IF-Anweisung wird der Befehlsdatei mitgeteilt, die Schleife jeweils erneut zu beginnen, solange der Wert in "Schleife{$$}" größer als 0 ist. Dadurch wird die Zeile "Schleifenanzahl" mit dem neuen Wert ausgegeben.

Die Schleife wird beendet, sobald der Wert in "Schleife{$$}" 0 ist. Vor Beenden der Befehlsdatei werden die beiden temporären Dateien gelöscht.

8.3.4 Verwenden von PIPE:

Soll die Auflistung des Inhalts eines Geräts einem anderen Prozeß bereitgestellt werden, gehen Sie wie folgt vor:

Prozeß 1:

```
1> LIST Work: TO PIPE: ALL
```

Prozeß 2:

```
2> TYPE pipe:
```

Sollen die Ergebnisse mehrerer C-Kompilierungen zusammengefaßt werden, geben Sie folgende Befehle ein:

```
1> sc >pipe:ll milk.c
1> sc >pipe:ll snap.c
1> sc >pipe:ll crackle.c
1> sc >pipe:ll pop.c
1> TYPE pipe:ll
```

Zur Verwendung von Kanalnamen geben Sie folgenden Befehl ein:

```
1> LIST >pipe:alles
```

Mit diesem Befehl wird die Auflistung an die Pipe "alles" gesendet.

```
1> COPY #?.c TO >pipe:all_c/32000
```

Mit diesem Befehl wird ein Kanal namens "all_c" und eine Puffergröße von 32000 Byte angegeben.

Soll die maximale Pufferanzahl 5 sein, geben Sie folgendes ein:

```
1> DIR >pipe://5
```

Mit diesem Befehl wird ein Kanal ohne Kanalnamen erstellt. Dabei sind maximal 5 Puffer zulässig.

8.3.5 Rekursive AmigaDOS-Befehlsdateien

Soll eine Befehlsdatei erstellt werden, mit der jeder beliebige AmigaDOS-Befehl für den Inhalt eines Verzeichnisses, einschließlich der Unterverzeichnisse und deren Inhalt, aufgerufen werden kann, geben Sie mit Eds (s. o.) die folgende Befehlsdatei ein und speichern Sie sie als "S:RPAT":

```
.KEY COM/A,PATH,OPT,RD
; Befehl als Argument COM, Pfad zum Verzeichnis-
;baum als PATH eingeben, der verarbeitet wird,
;falls nicht aktuelles Verzeichnis, ggf. Option
;für den Befehl als OPT eingeben. Für RD müssen
;Sie nichts eingeben.
;
.BRA {
.KET }
FAILAT 21      ; nicht abbrechen, falls LIST keine
               ;Elemente findet
; zunächst Dateien suchen, dann Skript generieren:
LIST >T:trf{$$} "{PATH}" FILES LFORMAT="{COM}
 *"%p%n*" *"{OPT}*"
IF EXISTS T:trf{$$}
    ; Dateien gefunden, neue Befehlsdatei aus-
    ; führen und "aufräumen"
    EXECUTE T:trf{$$}
    DELETE  T:trf{$$}
ENDIF
;
;Unterverzeichnisse suchen, für jedes RPAT
;aufrufen:
LIST >T:trd{$$}{RD} "{PATH}" DIRS LFORMAT="RPAT
 *"{COM}*" *"%p%n*" *"{OPT}*" RD=.{RD}"
;Mit dem Argument RD werden Punkte an die Namen
;der temporären Befehsldateien angehängt, um
;Verzeichnisebenen zu unterscheiden.
;
IF EXISTS T:trd{$$}{RD}
   ; Unterverzeichnisse vorhanden, neue Befehls-
   ; datei ausführen und "aufräumen"
   EXECUTE T:trd{$$}{RD}
   DELETE  T:trd{$$}{RD}
ENDIF
```

Sie müssen das Schutzbit "s" für die Befehlsdatei RPAT aktivieren. Außerdem muß sich die Datei bei Ausführung des Befehls RPAT im Suchpfad befinden. Vergleichbare Methoden können Sie den Befehlsdateien "S:SPAT" und "S:DPAT" entnehmen, die zusammen mit Ihrem System geliefert werden. Im Anhang B.2.2 finden Sie Näheres zu SPAT und DPAT.

Anhang A

Fehlermeldungen

Dieser Anhang enthält eine Liste der AmigaDOS-Fehler mit Angabe der wahrscheinlichen Ursachen und Vorschlägen zur Fehlerbehebung. Diese Fehlermeldungen werden vom System ausgegeben, wenn im Programm ein Fehler auftritt oder ein Befehl aufgrund eines Benutzerfehlers nicht ausgeführt wird. Fehlermeldungen unterscheiden sich von Dialogfenstern. Bei letzteren handelt es sich um Systemmeldungen, die die Eingabe bestimmter Korrekturen oder Änderungen ermöglichen, so daß das Programm, das Skript oder der Befehl weiter ausgeführt werden können. Werden in einem solchen Dialogfenster keine Eingaben gemacht, kommt es zu einem Fehler.

Fehler	Meldung	Wahrscheinliche Ursache	Empfohlene Maßnahme
103	Speicherplatzmangel	Der Amiga verfügt nicht über ausreichend Speicherplatz zum Ausführen dieser Operation. Der Speicher ist ggf. fragmentiert.	Schließen Sie alle nicht benötigten Anwendungen. Geben Sie den Befehl erneut ein. Tritt der Fehler weiterhin auf, starten Sie das System neu. Möglicherweise sollte man eineRAM-Erweiterung einbauen.
115	Ungültiger Zahlenwert	Das Programm erwartet ein numerisches Argument	Überprüfen Sie das Befehlsformat.

Fehler (Forts.)	Meldung	Mögliche Ursache	Empfohlene Maßnahme
116	Gefordertes Argument fehlt	Ein erforderliches Argument wurde nicht angegeben.	Überprüfen Sie das Befehlsformat.
117	Argument nach Schlüsselwort fehlt	Schlüsselwort wurde ohne Argument angegeben.	Überprüfen Sie das Befehlsformat.
118	Falsche Anzahl an Argumenten	Zu viele oder zu wenig Argumente.	Überprüfen Sie das Befehlsformat.
119	Ungerade Anzahl von Anführungszeichen	Die Anzahl der Anführungszeichen muß gerade sein.	Setzen Sie Anführungszeichen sowohl am Anfang als auch am Ende der Pfadangabe oder der Zeichenkette.
120	Argumentzeile ist ungültig oder zu lang	Die Befehlszeile ist inkorrekt oder enthält zu viele Argumente.	Überprüfen Sie das Befehlsformat.
121	Datei ist nicht ausführbar	Tippfehler im Befehlsnamen, oder die Datei ist keine ladbare (Programm- oder Skript-) Datei.	Geben Sie den Dateinamen nochmals ein und stellen Sie sicher, daß es sich um eine Programmdatei handelt. Zum Ausführen eines Skripts muß entweder das s-Bit gesetzt sein oder der Befehl EXECUTE verwendet werden.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
202	Objekt ist in Gebrauch	Die/das angegebene Datei/Verzeichnis wird bereits von einer anderen Anwendung belegt bzw. ist anderweitig zugeordnet.	Stoppen Sie die Anwendung, die die Datei oder das Verzeichnis benutzt bzw. heben Sie die Zuordnung auf.
203	Objekt existiert bereits	Es gibt bereits ein Vereichnis/eine Datei mit diesem Namen.	Verwenden Sie einen anderen Namen oder löschen Sie die/das existierende Datei/Vereichnis.
204	Verzeichnis nicht gefunden	AmigaDOS findet das angegebene Verzeichnis nicht.	Überprüfen Sie den Verzeichnisnamen mit dem Befehl DIR.
205	Objekt nicht gefunden.	AmigaDOS findet die angegebene Datei bzw. das angegebene Gerät nicht	Überprüfen Sie den Dateinamen mit dem Befehl DIR bzw. den Gerätenamen mit INFO.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
206	Ungültige Fenster-parameter	Dieser Fehler kann bei der Angabe der Größe für ein Shell-, ED- oder ICONX-Fenster auftreten. Möglicherweise ist das Fenster zu groß oder zu klein, oder Sie haben ein Argument weggelassen. Dieser Fehler kann auch beim Befehl NEWSHELL vorkommen, wenn der Name einer Einheit angegeben wird, die kein Fenster ist.	Geben Sie die Fensterangaben nochmals korrigiert ein.
209	Unbekannter DOS-Packet-Request-Typ	Sie haben von einem Geräte-Handler eine für diesen nicht mögliche Operation angefordert. Der Konsolen-Handler hat z. B. nicht die Möglichkeit, etwas umzubenennen.	Suchen Sie in den an die Geräte-Handler weitergeleiteten Requestcodes nach dem entsprechenden Request.
210	Ungültiger Objektname	Dateiname enthält ein ungültiges Zeichen oder ist zu lang.	Geben Sie den Namen nochmals ein und achten Sie auf gültige Zeichen und zulässige Länge.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
212	Objekt ist nicht vom geforderten Typ	Für eine Operation wurde statt des erforderlichen Dateinamens ein Verzeichnisname (oder umgekehrt) angegeben.	Überprüfen Sie den Namen und das Befehlsformat.
213	Disk ist nicht gültig	Wenn eben eine Disk eingelegt wurde, ist vielleicht nur die Disk-Überprüfung (Validierung) noch nicht abgeschlossen. Möglicherweise ist aber die Disk auch beschädigt.	Wenn eben erst eine Disk eingelegt wurde, warten Sie, bis die Überprüfung beendet ist. Dies dauert bei einer Diskette weniger als 1 Minute, bei einer Festplatte einige Minuten. Ist die Disk aber beschädigt, kann sie nicht gültig erklärt werden. Versuchen Sie in diesem Fall, die Dateien dieser Disk noch zu lesen und auf eine andere Disk zu kopieren.
214	Disk ist schreibgeschützt	Die Schreibschutzvorrichtung der Disk ist auf Schreibschutz eingestellt.	Entnehmen Sie die Disk, schieben Sie den Schreibschutzschieber in die andere Position und legen Sie die Disk wieder ein, oder verwenden Sie eine andere Disk bzw. benutzen Sie den Befehl LOCK OFF.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
215	Umbenennen auf anderen Datenträger versucht	Mit RENAME kann eine Datei von einem Verzeichnis in ein anderes gestellt werden, aber nicht von einem Datenträger auf einen anderen.	Versuchen Sie, die Datei mit COPY auf den Zieldatenträger zu kopieren. Löschen Sie sie danach bei Bedarf vom Quelldatenträger. Dann kann RENAME verwendet werden.
216	Verzeichnis ist nicht leer	Sie haben versucht, ein Verzeichnis zu löschen, das noch Dateien oder Unterverzeichnisse enthält.	Wenn Sie wirklich das komplette Verzeichnis löschen wollen, verwenden Sie die Option ALL des Befehls DELETE.
217	Zu tiefe Schachtelung	Die Verzeichnisverschachtelung ist zu tief.	Organisieren Sie Ihre Verzeichnisse neu oder wechseln Sie Verzeichnisse stufenweise, damit die gewünschte Ebene erreicht wird.
218	Gerät (oder Datenträger) ist nicht angemeldet	Ist das Gerät eine Diskette, wurde sie nicht ins Laufwerk eingelegt. Handelt es sich um ein anderes Gerät, wurde es nicht mit dem Befehl MOUNT angemeldet, oder sein Name wurde falsch geschrieben.	Legen Sie die richtige Diskette ein, melden Sie das Gerät an, prüfen Sie die Korrektheit des Gerätenamens oder korrigieren Sie die MountList /Mount-Datei bzw. weisen Sie den Gerätenamen entsprechend zu.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
219	Fehler bei Suchlesen	Beim Verarbeiten einer Datei trat ein Fehler auf.	Stellen Sie sicher, daß SEEK nur innerhalb einer Datei ausgeführt wird. Dateiübergreifendes Suchen ist mit SEEK nicht möglich.
220	Kommentar ist zu lang	Der Kommentar ist länger als das zulässige Maximum (79 Zeichen).	Verkürzen Sie den Kommentar.
221	Disk ist voll	Zur Ausführung der angeforderten Operation ist nicht genug freier Speicherplatz auf der Disk.	Löschen Sie einige nicht benötigte Dateien oder Verzeichnisse oder verwenden Sie eine andere Disk.
222	Objekt ist löschgeschützt	Das Schutzbit d (für "deletable" - löschbar) der Datei bzw. des Verzeichnisses ist gelöscht.	Wenn Sie die Datei oder das Verzeichnis tatsächlich löschen wollen, setzen Sie mit PROTECT das entsprechende d-Bit oder verwenden Sie die Option FORCE des Befehls DELETE.
223	Datei ist schreibgeschützt	Das Schutzbit w (für "writeable" - schreibbar) der Datei ist gelöscht.	Wenn Sie die Datei tatsächlich überschreiben wollen, setzen Sie mit PROTECT das entsprechende w-Bit.
224	Datei ist lesegeschützt	Das Schutzbit r (für "readable" - lesbar) der Datei ist gelöscht.	Setzen Sie mit PROTECT das r-Bit der Datei.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
225	Keine gültige DOS-Disk	Die Disk im Laufwerk ist keine AmigaDOS-Disk, nicht formatiert oder beschädigt.	Stellen Sie sicher, daß Sie die richtige Disk verwenden. Wenn Sie sicher sind, daß die Disk bisher funktioniert hat, verwenden Sie ein Disk-Wiederherstellungsprogramm, um die Dateien zu retten. Wenn die Disk noch nicht formatiert ist, holen Sie dies bitte mit FORMAT nach.
226	Keine Diskette im Laufwerk	Die Diskette ist nicht richtig in das angegebene Laufwerk eingelegt	Legen Sie die richtige Diskette in das angegebene Laufwerk ein.
232	Keine weiteren Verzeichniseinträge	Der AmigaDOS-Aufruf EXNEXT findet keine weiteren Einträge in dem Verzeichnis, das Sie gerade prüfen.	Beenden Sie das Aufrufen von EXNEXT.
233	Objekt im Verbund	Sie haben versucht, eine Operation an einem Verbundobjekt ("Soft Link") auszuführen, die nicht unterstützt wird.	Keine.
235	Ungültiger Hunk in zu ladender Datei	Das geladene Programm ist beschädigt.	Laden Sie eine neue Kopie oder das Originalprogramm.
241	Kollision bei Datensatzsperre	Eine andere Anwendung greift auf die Datenbank zu.	Versuchen Sie nochmals, auf die Datenbank zuzugreifen.

Fehler (Forts.)	Meldung (Forts.)	Mögliche Ursache (Forts.)	Empfohlene Maßnahme (Forts.)
242	Zeitüberschreitung bei Datensatzsperre	Der Datenbankeintrag wird durch eine andere Anwendung blockiert.	Versuchen Sie es erneut oder verlassen Sie die andere Anwendung und unternehmen Sie danach einen neuen Versuch.
303	Pufferüberlauf	Dieser Fehler tritt auf, wenn eine Zeichenkette für ein Namensmuster zu lang ist.	Verkürzen Sie die Zeichenkette.
304	***Abbruch	Dieser Fehler tritt auf, wenn das Programm durch Drücken der Tasten Ctrl - C gestoppt wurde.	Keine
305	Datei ist nicht ausführbar	Das Schutzbit e (für "executable" - ausführbar) der Datei ist gelöscht.	Siehe Fehler 121.



Anhang B
Zusätzliche Amiga-Verzeichnisse

Zusätzlich zu den AmigaDOS-Befehlen gibt es weitere Dateien und Verzeichnisse auf Ihrer Workbench-Disk. In diesem Kapitel werden folgende Verzeichnisse (Schubladen) beschrieben:

- DEVS:
- S:
- L:
- FONTS:
- LIBS:
- REXX:
- LOCALE:
- ENVARC:
- ENV:
- CLIPS:
- T:
- Classes
- C:

Die Schubladen von DEVS: werden im *Workbench-Benutzerhandbuch* beschrieben.

Sie müssen den Inhalt der hier aufgelisteten Verzeichnisse nicht genau verstehen. Falls nicht anders angegeben, können Sie diese Verzeichnisse ignorieren. Sie müssen jedoch ihren Verwendungszweck und Ihren Standort für den Fall kennen, daß Sie eine Datei in einem dieser Verzeichnisse versehentlich löschen oder umbenennen

oder es versäumen, erforderliche Elemente in ein dafür vorgesehenes Verzeichnis zu kopieren.

Abb. B-1 zeigt die auf Amiga-Festplattensystemen vorgegebene Verzeichnisstruktur. Verzeichnisse mit Piktogrammen, die auf der Workbench (Schubladen) sichtbar sind, befinden sich auf der linken Seite der Auflistung, die anderen Verzeichnisse auf der rechten Seite. Der vorgebene Inhalt und die vorgegebene Struktur kann Abweichungen von der Darstellung aufweisen, da ggf. firmenseittig Ressourcen hinzugefügt, geändert oder entfernt werden.

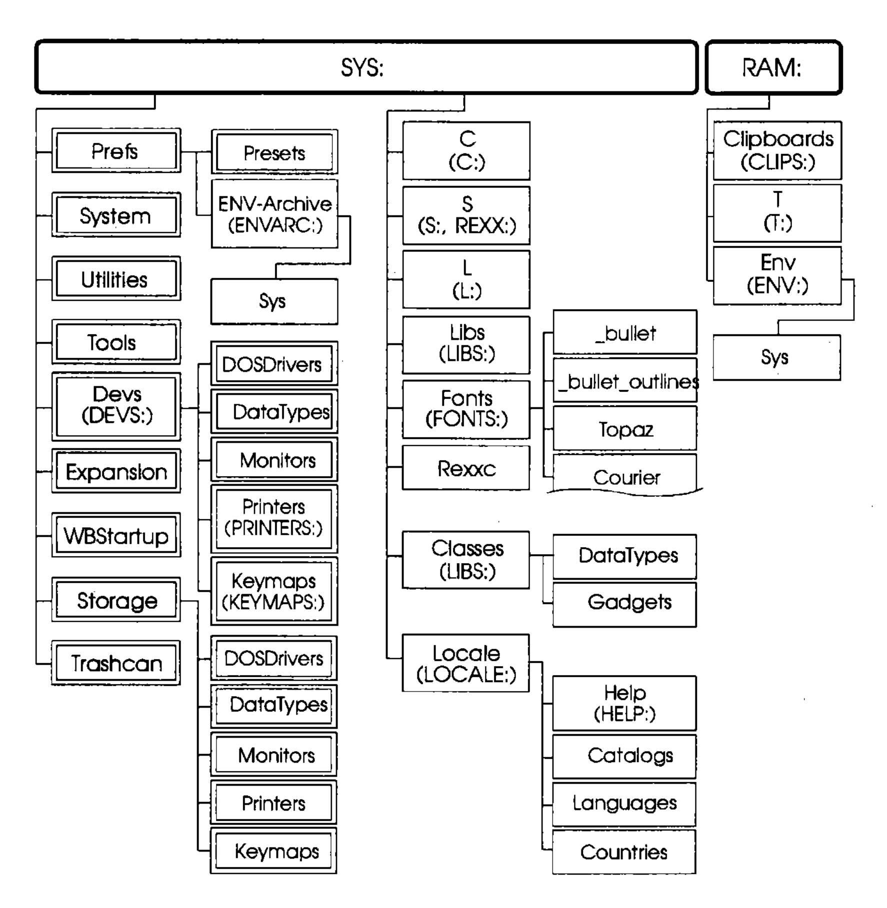


Abb. B-1. Verzeichnisstruktur bei Festplattensystemen

Hinweis Bei den meisten Diskettensystemen sind die Verzeichnisse LOCALE: (Landeseinstellungen), FONTS: (Zeichensätze) und "Storage" (Vorrat) auf getrennten Datenträgern gespeichert.

Ein Großteil dieser Verzeichnisse wird automatisch dem Datenträger SYS: oder der RAM-Disk zugewiesen. Falls erforderlich, können Sie diese Verzeichnisse sowie SYS: mit dem Befehl ASSIGN auch anderen Datenträgern zuweisen.

So können Sie etwa FONTS: einer bestimmten Disk zuweisen, z. B. "FontDisk:". Viele Anwendungen suchen automatisch im Verzeichnis FONTS: nach den benötigten Zeichensätzen - unabhängig davon, welcher Disk dieses Verzeichnis zugeordnet ist. Durch Änderung der Zuweisung von FONTS: können Sie Anwendungen den Zugriff auf die Zeichensätze von "FontDisk:" ermöglichen.

B.1 DEVS: (Geräte)

Neben den Schubladen "DOSDrivers" (DOS-Treiber), "Keymaps" (Tastaturbelegungen), Printers (Drucker), Monitors (Monitore) und DataTypes (Datentypen), die im *Workbench-Benutzerhandbuch* beschrieben werden, enthält die Schublade DEVS: Dateien und Unterverzeichnisse für die Geräte, die an den Amiga angeschlossen werden können.

Auf die Schubladen "DEVS:Keymaps" und "DEVS:Printers" können Sie über deren zugewiesene Namen KEYMAPS: und PRINTERS: verweisen.

B.1.1 Gerätedateien

Die folgende Liste zeigt die .device-Dateien in der Schublade "DEVS:" und ihre jeweiligen Funktionen:

clipboard.device Steuert den Zugriff auf CLIPS: (den Zwischenspeicher).

parallel.device Steuert den Zugriff auf die parallele Schnittstelle.

printer.device	Steuert den Zugriff auf die Druckereinheit.
serial.device	Steuert den Zugriff auf die serielle Schnittstelle.
mfm.device	Steuert den Zugriff auf MS-DOS-Disketten mit Hilfe von CrossDOS.

Nähere Informationen zu den .device-Dateien finden Sie in den ROM-Kernel-Handbüchern (englisch) von Addison-Wesley (*ROM Kernel Reference Manuals).*

B.1.2 Weitere Dateien

Die Schublade DEVS: enthält folgende zusätzliche Dateien:

system-configuration	Enthält Konfigurationsdaten der Voreinsteller-Editoren, die beim Starten des Systems aktiviert werden.
postscript_init.ps	Enthält die Daten, die bei Verwendung des PostScript-Drucker-Voreinstellers (PrinterPS) zum Initialisieren eines PostScript-Druckers erforderlich sind.

B.1.2.1 Verwenden von Anmeldedateien oder einer MountList

Damit der Amiga auf neue Geräte zugreifen kann, muß er über neu hinzugefügte Geräte informiert werden. Dabei kann es sich sowohl um physische Geräte (z. B. ein Bandlaufwerk) als auch um logische Geräte (Software; z. B. eine resetfeste RAM-Disk) handeln. Der Amiga kann mit folgenden Methoden informiert werden:

- Kopieren Sie den erforderlichen Treiber in die Schublade "Expansion" (Erweiterung). Das muß eine Datei mit Piktogramm (mit .info-Datei) sein, und in der Startup-Sequence muß der Befehl BINDDRIVERS aufgerufen werden.
- Kopieren Sie die erforderliche Anmeldedatei in die Schublade "DOSDrivers" (DOS-Treiber).
- Fügen Sie den erforderlichen Eintrag für das Gerät in der Datei "MountList" hinzu und benutzen Sie den Befehl MOUNT.

Eine Anmeldedatei steht für ein Gerät, einen Handler oder ein Dateisystem. Für die Standardgeräte sind in der Schublade "DOSDrivers" in der Schublade DEVS: die zugehörigen Anmeldedateien mit Piktogrammen enthalten. Diese Geräte werden automatisch aus der Startsequenz heraus angemeldet. Von den anderen Geräten wird bei Aufruf des Befehls MOUNT ein Eintrag in einer MountList-Datei gelesen, der die Merkmale für das jeweilige Gerät festlegt.

Ab Version 2.1 der Amiga-Systemsoftware ist es aufgrund der Methode mit den Anmeldedateien nicht mehr erforderlich, eine MountList-Datei und den Befehl MOUNT zu verwenden. Anstelle der MountList-Datei mit Einträgen für alle anzumeldenden Geräte enthält die Schublade DEVS: jetzt die Schublade "DOSDrivers", die eine separate Anmeldedatei bzw. einen DOS-Treiber für jedes anzumeldende Gerät enthält. Der Inhalt einer Anmeldedatei stimmt im wesentlichen mit einem einzelnen Eintrag in der MountList-Datei überein.

Sie können jedoch weiterhin MountList-Dateien zum Anmelden von Geräten verwenden. Kopieren Sie dazu die MountList-Datei in die Schublade "DEVS:" und entfernen Sie alle DOS-Treiber, deren Dateinamen mit den Namen der MountList-Einträge übereinstimmen, aus der Schublade "DOSDrivers". (Eine Anmeldedatei setzt einen MountList-Eintrag mit demselben Namen außer Kraft.)

B.1.2.2 Erstellen einer Anmeldedatei bzw. eines MountList-Eintrags

Die folgenden Erläuterungen zu Anmeldedateien gelten, falls nicht anders angegeben, auch für MountList-Dateien.

Anmeldedateien enthalten Schlüsselwörter, die das Gerät, den Handler oder das Dateisystem näher beschreiben, sowie die Werte für diese Schlüsselwörter. Bestimmte Schlüsselwörter gelten möglicherweise nur für Dateisysteme oder Handler. Wird ein Schlüsselwort nicht angegeben, verwendet das System einen Standardwert. Überprüfen Sie stets den jeweiligen Standardwert, um sicherzustellen, daß er für das jeweilige Gerät geeignet ist.

Beim Erstellen von Anmeldedateien und MountList-Dateien müssen Sie folgende Regeln beachten:

- Die Datei muß als ASCII-Klartextdatei erstellt werden.
- Der Name einer Anmeldedatei muß mit dem Gerätenamen übereinstimmen; MountList-Dateien sollten den Namen "MountList" erhalten.
- Jeder Eintrag in einer MountList-Datei muß mit dem Gerätenamen beginnen. Bei Mount-Dateien ist der Gerätename nicht anzugeben.
- Auf Schlüsselwörter muß ein Gleichheitszeichen (=) folgen.
- Schlüsselwörter sind durch ein Semikolon voneinander zu trennen oder in jeweils separate Zeilen zu schreiben.
- Kommentare sind in der für C üblichen Schreibweise zulässig (d. h. sie müssen mit /* beginnen und auf */ enden).
- Jeder MountList-Eintrag muß mit dem Zeichen # enden. Bei Anmeldedateien ist dieses Zeichen nicht anzugeben.

Wenn Sie eine neue Anmeldedatei bzw. einen neuen MountList-Eintrag erstellen, lesen Sie zunächst die Informationen in der Dokumentation, die zusammen mit dem Gerät geliefert wurde, oder gehen Sie von einem Eintrag für ein vergleichbares Gerät aus. Falls erforderlich, ändern Sie einige Schlüsselwörter oder fügen notwendige Schüsselwörter hinzu.

Die folgende Tabelle zeigt die Schlüsselwörter und die Standardwerte, die in einer Anmeldedatei bzw. einer MountList-Datei zulässig sind, sowie ihre Funktion. Die Standardwerte stehen in spitzen Klammern.

Schlüsselwort	Funktion
Handler=<keiner>	Handler-Eintrag (z. B. Handler = L:queue-handler).
EHandler=<keiner>	Eintrag für einen Umgebungs-Handler.
FileSystem=<keiner>	Eintrag für ein Dateisystem (z. B. FileSystem = L:CrossDOSFileSystem).

Schlüsselwort (Forts.)	Funktion (Forts.)
Device=<keiner>	Eintrag für ein Gerät (z. B. Device = DEVS:mfm.device). Dieses Argument ist zum Anmelden eines Dateisystems erforderlich. Für "Device" gibt es keinen Standardwert. Sie müssen einen Wert angeben.
Priority=<10>	Die Prozeßpriorität; 5 eignet sich für Handler, 10 für Dateisysteme.
Unit=<0>	Einheitennummer des Geräts (z. B. 0 für PC0:).
Flags=<0>	Kennzeichen für OpenDevice (in der Regel 0).
Surfaces=<keiner>	Anzahl der Oberflächen (2 bei Diskettenlaufwerken, bei Festplatten unterschiedlich). Dieses Argument ist zum Anmelden eines Dateisystems erforderlich. Für "Surfaces" gibt es keinen Standardwert. Sie müssen einen Wert angeben.
SectorsPerBlock=<keiner>	Anzahl der phyischen Disk-Sektoren in jedem logischen Block, der von einem Dateisystem verwendet wird.
SectorsPerTrack=<keiner>	Anzahl der Blöcke pro Spur. Dieses Argument ist zum Anmelden eines Dateisystems erforderlich. Für "SectorsPerTrack" gibt es keinen Standardwert. Sie müssen einen Wert angeben.
SectorSize=<512>	Anzahl der Bytes in einem Block auf dem Gerät. Die meisten Geräte arbeiten mit 512-Byte-Blöcken, einige mit anderen Größen (CD-ROMs verwenden z. B. 2048-Byte-Blöcke).
Reserved=<2>	Anzahl der für den Boot-Block reservierten Blöcke; es empfiehlt sich der Wert 2
Interleave=<0>	Sektorversatz; variiert von Gerät zu Gerät.
LowCyl=<keiner>	Zu verwendender Startzylinder. Dieses Argument ist zum Anmelden eines Dateisystems erforderlich. Für "LowCyl" gibt es keinen Standardwert. Sie müssen einen Wert angeben.

Schlüsselwort (Forts.)	Funktion (Forts.)
HighCyl=<keiner>	Zu verwendender Endzylinder. Dieses Argument ist zum Anmelden eines Dateisystems erforderlich. Für "HighCyl" gibt es keinen Standardwert. Sie müssen einen Wert angeben.
Stacksize=<600>	Die dem Prozeß zugeordnete Stackgröße.
Buffers=<5>	Anzahl anfänglicher Cache-Puffer von je 512 Byte Größe. Falls ausreichend Kapazität im RAM verfügbar ist, erhöhen Sie den Wert zur Leistungsverbesserung auf einer Disk.
BufMemType=<3>	Für Puffer verwendeter Speichertyp; (0 und 1 = Beliebig, 2 und 3 = Chip, 4 und 5 = Fast).
Mount=<0>	Siehe Beschreibung des Schlüsselworts ACTIVATE (Synonym des Schlüsselworts MOUNT).
MaxTransfer= <0x7ffffff>	Maximale Anzahl Bytes, die gleichzeitig von einem beliebigen Dateisystem übertragen werden können. Verwenden Sie kleine Werte (z. B. 0xffff) für "MaxTransfer", um Kompatibilität zu älteren Festplattensystemen zu erreichen.
Mask=<0xfffffffe>	Adreßmaske zur Angabe des Speicherbereichs, der für DMA-Transfers in beliebigen Dateisystemen verwendet werden kann. Verwenden Sie "Mask", um Kompatibilität zu älteren Festplattensystemen zu erreichen.
GlobVec=<2>	Globaler Vektor für den Prozeß; -1 bedeutet kein Globalvektor (bei C- und Assembler-Programmen), 0 richtet einen privaten Globalvektor (GV) ein. Wenn dieses Schlüsselwort fehlt, wird der Globalvektor für gemeinsame Benutzung verwendet. Bei Geräten mit Amiga-Dateisystem dürfen Sie dieses Schlüsselwort nicht verwenden.
Startup=<keiner>	Zeichenkette, die dem Gerät, Handler oder Dateisystem beim Systemstart als BPTR auf einen BSTR übergeben wird.

Schlüsselwort (Forts.)	Funktion (Forts.)
Activate=<0>	Bei Angabe eines positiven Werts wird das Gerät bzw. der Handler durch ACTIVATE nicht erst beim ersten Zugriff geladen, sondern sofort. Synonym zu MOUNT.
BootPri=<0>	Wert, der für ein Gerät, von dem gestartet oder das angemeldet werden kann, eine Start-Priorität setzt. Mögliche Werte sind -129 bis 127. Vereinbarungsgemäß bedeutet -129, daß das Gerät nicht startfähig ist und nicht automatisch angemeldet wird.
DosType=<0x444F5300>	Bezeichnet den Dateisystemtyp. Dabei sind die hexadezimalen ASCII-Codes für drei Buchstaben und eine abschließende Nummer anzugeben. Diese Werte haben folgende Bedeutung:

Wert	ASCII	Dateisystem
0x444F5300	DOS0	Original (OFS)
0x444F5301	DOS1	FastFileSystem (FFS)
0x444F5302	DOS2	OFS (Internationaler Modus)
0x444F5303	DOS3	FFS (Internationaler Modus)
0x444F5304	DOS4	Verzeichnis-Cache OFS (Internationaler Modus)
0x444F5305	DOS5	Verzeichnis-Cache FFS (Internationaler Modus)
0x4D534400	MSD0	MS-DOS

Schlüsselwort (Forts.)	Funktion (Forts.)
Baud=<1200>	Baudrate eines seriellen Geräts.
Control=<0>	Wortlänge, Parität und Stoppbits für serielle Geräte.

Schlüsselwort (Forts.)	Funktion (Forts.)
Forceload=<0>	Erzwingt, daß ein Dateisystem von Disk geladen wird, selbst wenn die Ressourcenliste einen geeigneten Eintrag enthält. Bei Angabe von 0 (Standardwert) wird die Ressourcenliste vor der Disk durchsucht. Bei Angabe von 1 wird stets von der Disk geladen.

B.2 Verzeichnis S:

Das Verzeichnis S: ist im allgemeinen für AmigaDOS- und ARexx-Befehlsdateien reserviert. Sie können jedoch auch andere Dateien im Verzeichnis S: speichern und Befehlsdateien in andere Verzeichnisse stellen. Das Verzeichnis S: enthält neben der Datei "Startup-sequence" (Startsequenz), der ggf. von Ihnen erstellten Datei "User-startup" (Benutzerstart) und der Datei "Shell-startup" (Shell-Start) die in diesem Abschnitt beschriebenen Befehlsdateien.

B.2.1 ED-Startup

Diese Datei enthält ED-Befehle zur Konfiguration des Texteditors ED. Unter anderem ist sie für die Standardbelegung der Funktionstasten zuständig. Die Datei kann zur Anpassung der Tastenzuordnung ediert werden. Wenn Sie diese Datei entfernen oder umbenennen, werden die erweiterten ED-Menüs mit zusätzlichen Befehlen aktiviert (siehe Kapitel 4).

Andere Dateien, die ED-Befehle enthalten, können in S: gespeichert werden, damit sie mit dem ED-Schlüsselwort WITH zur Ausführung eigener Edieroperationen verwendet werden können.

B.2.2 SPat, DPat

Diese Befehlsdateien erweitern Befehle um Namensmusterfunktionen, wenn diese normalerweise nicht darüber verfügen. Wenn diese Befehlsdateien zusammen mit einem Befehl als Argument aufgerufen werden, benutzen "SPat" und "DPat" den Befehl LIST zur Erstellung temporärer Befehlsdateien im Verzeichnis T:. Anschließend werden diese Befehlsdateien aufgerufen. "SPat" und "DPat" können auch innerhalb von Alias-Namen für Befehle verwendet werden.

"SPat" erweitert Befehle mit einem einzelnen Argument um die Möglichkeit der Namensmusterauswertung. Soll z. B. der Texteditor ED zum Edieren aller Dateien im Verzeichnis S: verwendet werden, die mit dem Buchstaben "s" beginnen, geben Sie folgendes ein:

```
1> SPat ED S:s#?
```

Es wird eine Befehlsdatei erzeugt, die ungefähr wie folgt aussieht:

```
ED "s:Shell-startup"
ED "s:SPat"
ED "s:Startup-sequence"
```

"SPat" ruft dann diese Befehlsdatei auf, d. h. ED wird zur Bearbeitung der entsprechenden Dateien dreimal aufgerufen.

"DPat" erweitert Befehle mit zwei Argumenten um die Möglichkeit der Namensmusterauswertung. Hinter "DPat" und dem Befehlsnamen geben Sie die beiden Argumente (getrennt durch ein Leerzeichen) ein. Verwenden Sie dabei die Jokerzeichen, die für die gewünschten Namensmuster erforderlich sind.

B.2.3 PCD

Wie beim Befehl CD wird durch die Befehlsdatei PCD das aktuelle Verzeichnis gewechselt. Darüber hinaus "merkt" sich die Befehlsdatei PCD jedoch das Verzeichnis, von dem aus gewechselt wurde. Auf diese Weise können Sie ohne Eingabe des vollständigen Pfads zum vorherigen aktuellen Verzeichnis zurückkehren. Jede Shell verfügt über einen unabhängigen PCD-Speicher.

Wenn Sie PCD in einer bestimmten Shell zum ersten Mal verwenden, müssen Sie den vollständigen Pfad zu einem Verzeichnis eingeben.

Wenn Sie PCD mit dem Pfadargument eingeben, hat der Befehl denselben Effekt wie der Befehl CD, d. h. das im Pfad angegebene Verzeichnis wird zum aktuellen Verzeichnis gemacht. Gleichzeitig wird jedoch der Pfad zu dem Verzeichnis gespeichert, von dem aus gewechselt wird. Anschließend können Sie das aktuelle Verzeichnis mit den üblichen Methoden wechseln. Soll später zum anfänglichen Verzeichnis zurückgekehrt werden, geben Sie PCD ohne Argumente ein. Beispiel:

```
1.System:> PCD Arbeit:Bild/24bit
1.Arbeit:Bild/24bit> PCD
1.System:>
```

Wenn Sie die Befehlsdatei PCD später erneut aufrufen, wird jeweils zum Verzeichnis gewechselt, in dem der Befehl PCD zuletzt eingegeben wurde:

```
1.System:> PCD
1.Arbeit:Bild/24bit> /
1.Arbeit:Bild> PCD
1.System:>
```

PCD verwendet zum Speichern des "gemerkten" Verzeichnisses die Zuweisungsliste. Bei Verwendung von PCD enthält die Liste die Zuweisung **from<n>:**. Dabei steht <n> für die Shell-Nummer. Wenn Sie den Befehl PCD bei dessen erster Verwendung ohne Pfadargument eingeben, wird eine Fehlermeldung angezeigt.

Kapitel 8 enthält weitere Beispiele zur Verwendung des Befehls PCD.

B.3 Verzeichnis L:

Dieses Verzeichnis enthält die Geräte-Handler. Bei diesen handelt es sich um Softwaremodule, die als Zwischenstufe zwischen AmigaDOS und den vom Amiga verwendeten Geräten dienen. Die meisten Handler werden allerdings so behandelt, als wären sie tatsächliche, physische Geräte. Sie werden auch mit ihrem Gerätenamen angesprochen.

Handler müssen in den Anmelde- bzw. MountList-Dateien für ihre jeweiligen Geräte benannt sein. Handler werden im allgemeinen nicht vom Benutzer direkt aufgerufen oder manipuliert, sondern von Programmen. Zum Lieferumfang einiger Geräte oder Anwen-

dungsprogramme gehören eigene Handler. Diese müssen ebenfalls in das Verzeichnis L: kopiert werden.

B.3.1 Aux-Handler

Der Aux-Handler ermöglicht ungepufferte serielle Ein- und Ausgabe. Er ist in erster Linie ein Konsolen-Handler, der nicht den Bildschirm und die Tastatur des Amiga verwendet, sondern die serielle Schnittstelle.

Die DOSDrivers-Anmeldedatei für AUX lautet:

```
Handler = L:Aux-handler
Stacksize = 1000
Priority = 5
```

Mit dem Aux-Handler können Sie an Ihrem Computer ein serielles Terminal benutzen. Beispiel:

```
1> NEWSHELL AUX:
```

B.3.2 Queue-Handler (PIPE:)

Der Queue-Handler ist ein E/A-Mechanismus für die E/A-Kommunikation zwischen Programmen. Über den Queue-Handler wird ein Kanal namens PIPE: für die Kommunikation zwischen Prozessen erstellt. Weitere Informationen zu PIPE: können Sie Anhang D entnehmen.

B.3.3 Port-Handler

Der Port-Handler ist die AmigaDOS-Schnittstelle zu den Geräten SER:, PAR: und PRT:.

Für den Zugriff auf SER: können Sie Einstellungen für die Baudrate und Steuerinformationen angeben. Das zugehörige Format lautet SER:<Baud/Steuerung>. Dabei steht "Baud" für die Baudrate (numerischer Wert) und "Steuerung" für eine Zeichenfolge aus drei Zeichen. Die einzelnen Zeichen haben folgende Bedeutung:

Bits pro Zeichen	Erstes Zeichen, entweder 7 oder 8.
Parität	Zweites Zeichen, N (keine Parität - no parity), O (ungerade Parität - odd parity), E (gerade Parität - even parity), M (Markierungsparität - mark parity), S (Leer-Parität - space parity)
Anzahl der Stoppbits	Drittes Zeichen, entweder 1 oder 2.

Beispiel:

```
SER:9600/8N1
```

Bei dieser Angabe wird die Verbindung zur seriellen Schnittstelle hergestellt. Dabei wird die Baudrate auf 9600 eingestellt. Die Daten werden im 8-Bit-Format dargestellt. Es ist kein Paritätsbit und ein Stoppbit vorhanden.

Wenn Sie beim Zugriff auf SER: keinen Wert für die Baudrate oder die Steuerung angeben, werden die Einstellungen aus dem Seriell-Voreinsteller (Serial) verwendet.

B.3.4 CrossDOSFileSystem

Das CrossDOSFileSystem ist zur Verwendung von CrossDOS erforderlich.

B.3.5 FileSystem_Trans

Das Verzeichnis "FileSystem_Trans" enthält folgende Dateien, die für die CrossDOS-Textumsetzung erforderlich sind:

DANSK.crossdos	Dient als Filter für dänische Textdateien.
INTL.crossdos	Konvertiert internationale Zeichen.
MAC.crossdos	Konvertiert ASCII-Dateien des Apple Macintosh.

B.3.6 CDFileSystem

Das CDFileSystem ist zur Verwendung von CD-ROM-Laufwerken erforderlich.

B.4 FONTS:

Die FONTS:-Zuweisung bezieht sich auf die Disk oder das Verzeichnis, das die Daten aller auf dem Amiga verfügbaren Schriftattribute (Styles) für alle Zeichensätze (Fonts) enthält. Bitmap- und Umriß-Zeichensätze werden unterschiedlich gespeichert.

B.4.1 Bitmap-Zeichensätze

Für jeden Bitmap-Zeichensatz gibt es ein Unterverzeichnis und eine .font-Datei. Das Unterverzeichnis enthält Dateien für die unterschiedlichen Punktgrößen, mit denen der Zeichensatz zur Verfügung steht. Jede der Dateien enthält die Bitmap-Darstellung jedes Zeichens des Zeichensatzes mit der angegebenen Punktgröße.

So gibt es z. B. für den Zeichensatz "Emerald" ein Verzeichnis "Emerald" und die Datei "Emerald.font". Das Verzeichnis "Emerald" enthält zwei Dateien: 17 und 20. Diese Dateien enthalten die Daten, die für Emerald-Zeichensätze mit Größe 17-Punkt bzw. 20-Punkt benötigt werden. Die Datei "Emerald.font" enthält die Liste aller verfügbaren Punktgrößen und Schriftattribute (z. B. Fett, Kursiv usw.) zum jeweiligen Zeichensatz.

Viele Textverarbeitungsprogramme und Desktop-Publishing-Programme bringen zusätzliche Zeichensätze mit, die Sie in Ihr Verzeichnis FONTS: kopieren sollten. Immer wenn Sie einen neuen Zeichensatz in FONTS: aufnehmen, müssen Sie anschließend das Programm "FixFonts" aufrufen, um die .font-Datei für den neu hinzugekommenen Zeichensatz zu aktualisieren.

Der Zeichensatz "Topaz" ist der vom Amiga standardmäßig verwendete Zeichensatz. Er steht nicht nur im Verzeichnis FONTS:, sondern "Topaz 8" und "Topaz 9" sind auch im ROM gespeichert. Der Amiga kann folglich selbst dann noch Text anzeigen, wenn Sie versehentlich den gesamten Inhalt des Verzeichnisses FONTS: löschen sollten. "Topaz 11" findet sich jedoch nur im Verzeichnis "Topaz".

B.4.2 Umriß-Zeichensätze

Vom Umriß-Zeichensatzsystem Compugraphic Intellifont, das vom Amiga verwendet wird, werden die Zeichensatzdaten mit einer anderen Methode gespeichert. Wie bei Bitmap-Zeichensätzen gibt es für jede Schriftart eine .font-Datei. Darüber hinaus verfügt jede Schriftart aber über eine .otag-Datei. Die beiden Unterverzeichnisse "_bullet" und "_bullet_outlines" enthalten die eigentlichen Umrißinformationen für die auf Ihrem System installierten CG-Schriftarten. Edieren Sie diese Dateien niemals direkt. Verwenden Sie zur Verwaltung von Umriß-Zeichensätzen stets das Programm "Intellifont" oder die Installationsprogramme der jeweiligen Anwendung.

Hinweise Nicht auf allen Amiga-Systemen sind Umriß-Zeichensätze installiert.

B.5 Verzeichnis LIBS:

LIBS: enthält Bibliotheken (Libraries) mit Sammlungen von verschiedenen Softwareroutinen und mathematischen Funktionen, die in der Regel vom Betriebssystem und von Ihren Anwendungen verwendet werden. Im Verzeichnis LIBS: befinden sich folgende Dateien:

.library-Datei	Funktion
amigaguide.library	Funktionen, die vom AmigaGuide-Hypertextsystem verwendet werden
asl.library	Requester-Funktionen (Auswahlfenster) für Datei-, Zeichensatz- und Bildschirmmodus
bullet.library	Funktionen zum Suchen und Laden von Umriß-Zeichensätzen

.library-Datei (Forts.)	Funktion (Forts.)
commodities.library	Funktionen, die von Commodities-Exchange-Programmen verwendet werden
datatypes.library	Funktionen zur Bearbeitung der Dateitypen in "DEVS:DataTypes"
diskfont.library	Funktionen zum Suchen und Laden von Zeichensatzdateien
iffparse.library	Funktionen zum Lesen von IFF-Dateien
locale.library	Funktionen zur Verwendung landesspezifischer Anpassungen des Amiga-Betriebssystems
lowlevel.library	Bibliothek zur leichteren Programmierung von Spielen
mathieeedoubbas.library	IEEE-Funktionen mit Mathematik-Routinen für grundlegende Rechenfunktionen (Addition, Subtraktion usw.) in doppelter Genauigkeit
mathieeedoubtrans.library	IEEE-Funktionen mit Mathematik-Routinen für transzendente Rechenfunktionen (Sinus, Kosinus usw.) in doppelter Genauigkeit
mathieeesingtrans.library	Schnelle IEEE-Funktionen mit Mathematik-Routinen in einfacher Genauigkeit
mathtrans.library	FFP-Funktionen mit Mathematik-Routinen für transzendente Funktionen
realtime.library	Funktionen zum Synchronisieren von Multimediaeffekten, z. B. Musik und Animationen
rexxsupport.library	Von ARexx verwendete Funktionen
rexxsyslib.library	Hauptsächlich verwendete ARexx-Funktionen
version.library	Enthält Daten zur aktuellen Softwareversion und zum Stand der Überarbeitung
68040.library	Funktionen, die für Amiga-Systeme mit Mikroprozessor-Chip 68040 erforderlich sind

B.6 REXX:

REXX: ist ein weiterer Name, der dem Verzeichnis SYS:S zusätzlich zu S: zugewiesen wird. Wenn Sie von Ihnen geschriebene ARexx-Programme im Verzeichnis REXX: speichern, können Sie die Programme ohne Eingabe des vollständigen Pfads starten. Erstellen Sie ein neues Verzeichnis und weisen Sie REXX: auf dieses zu, wenn ARexx-Programme von AmigaDOS-Befehlsdateien getrennt gehalten werden sollen.

B.7 LOCALE:

Im Verzeichnis LOCALE: sucht der Amiga nach Sprachdateien und landesspezifischen Dateien, wenn Text in einer anderen Sprache als Englisch angezeigt werden soll. Bei Diskettensystemen entspricht das Verzeichnis LOCALE: der Diskette "Locale". Auf Festplattensystemen steht LOCALE: für das Verzeichnis "SYS:Locale".

LOCALE: enthält vier Verzeichnisse:

Countries (Länder)	Enthält jeweils eine .country-Datei für alle verfügbaren Länder. Diese Dateien enthalten landesspezifische Informationen wie z. B. Format von Datum und Uhrzeit oder Währungssymbole.
Languages (Sprachen)	Enthält .language-Dateien für alle verfügbaren Sprachen. Diese Dateien enthalten sprachspezifische Informationen wie z. B. die Namen der Wochentage.
Catalogs (Kataloge)	Enthält Unterverzeichnisse für alle verfügbaren Sprachen. In jedem dieser Unterverzeichnisse befindet sich ein Verzeichnis "Sys". Dieses Verzeichnis enthält die Übersetzungen in den jeweiligen Sprachen. "Catalogs/<Sprache>/Sys" enthält Menü-, Feld- und Meldungstexte für die jeweilige Sprache.
Help (Hilfe; HELP:)	Enthält Unterverzeichnisse für alle verfügbaren Sprachen. In jedem dieser Unterverzeichnisse befindet sich ein Verzeichnis "Sys". Dieses Verzeichnis enthält die Übersetzungen in den jeweiligen Sprachen. "Help/<Sprache> /Sys" ist für die AmigaGuide-Hilfstextdateien aller Anwendungen reserviert, die über AmigaGuide-Hilfstexte in der jeweiligen Landessprache verfügen.

B.8 ENVARC:

ENVARC: ist der dem Verzeichnis "SYS:Prefs/Env-Archive" zugewiesene Name. ENVARC: enthält stets ein Unterverzeichnis "Sys". In diesem Unterverzeichnis befinden sich die .prefs-Dateien, die von den Voreinsteller-Editoren erstellt werden, wenn Sie eine Einstellung ändern und "Speichern" auswählen. Mit dem Programm "IconEdit" erstellte benutzerspezifische Standardpiktogramme werden ebenfalls in diesem Unterverzeichnis gespeichert. (Wenn Sie benannte Einstellungen in einem Voreinsteller-Fenster über den Menüpunkt "speichern als" speichern, werden diese standardmäßig in die Schublade "Prefs/Presets" (Voreinsteller-Editoren/Voreinstellungen) gestellt.

Von anderen Workbench-Programmen, die Ihnen das Speichern von Konfigurationseinstellungen ermöglichen (z. B. MultiView), werden die zugehörigen Dateien ebenfalls in das Verzeichnis ENVARC: gestellt.

B.9 ENV:

ENV: steht für das Verzeichnis "RAM:Env", in das der Inhalt von ENVARC: während des Startvorgangs kopiert wird. In ENV: sind die Einstellungen der Voreinsteller-Editoren gespeichert, die durch Auswahl von "Speichern" oder "Benutzen" aktiviert wurden. Darüber hinaus befinden sich in diesem Verzeichnis die globalen Umgebungsvariablen (in Form von kleinen Textdateien).

B.10 CLIPS:

Dem Verzeichnis RAM:Clipboards ist der Name CLIPS: zugewiesen. In diesem Verzeichnis befinden sich die Daten, die durch Auswahl der Menüpunkte "Ausschneiden" (Cut) oder "Kopieren" (Copy) im Menü "Edieren" oder "Bearbeiten" (Edit) eines Programms in den Zwischenspeicher kopiert wurden.

B.11 T:

T: steht für das Verzeichnis RAM:T. Dieses Verzeichnis wird von einigen Befehlsdateien und Befehlen zum Speichern verschiedener temporärer Dateien verwendet.

B.12 Classes

Im Verzeichnis "Classes" werden die Daten zu objektorientierten Funktionen der Workbench (z. B. DataTypes) gespeichert, auf denen das Programm "MultiView" basiert. "Classes" enthält die Verzeichnisse "DataTypes" und "Gadgets" .

B.13 C:

C: steht für das Verzeichnis SYS:C, in dem alle nicht internen AmigaDOS-Befehle gespeichert werden. Dieses Verzeichnis ist stets im Suchpfad enthalten.

Anhang C

Reine Diskettensysteme

Die Einschränkungen, die beim Arbeiten mit Systemen mit Diskettenlaufwerk (ohne Festplatte) gelten, und das Wechseln von Disketten sind im *Workbench-Benutzerhandbuch* beschrieben. Dieser Anhang erläutert, wie AmigaDOS verwendet werden kann, um das Wechseln von Disketten auf ein Minimum zu reduzieren und den Arbeitsspeicher zu maximieren. Dazu dienen folgende Maßnahmen:

- Befehle resident machen
- Vorab Ressourcen in den Hauptspeicher laden
- Pfad zuordnen
- Dateien von der Workbench-Diskette löschen
- Verwenden der Ram-Disk
- Verwenden von RAD:

C.1 Befehle resident machen

Bei Systemen mit einem Diskettenlaufwerk empfiehlt es sich, die Befehle resident zu machen, die Sie am häufigsten benötigen, um schnell auf sie zugreifen zu können. Residente Befehle, stehen im Hauptspeicher des Amiga; um einen residenten Befehl aufzurufen, muß nicht vorher jedesmal die Workbench-Diskette eingelegt werden. Zum Kopieren eines Befehls in den Hauptspeicher des Amiga verwenden Sie den Befehl RESIDENT.

Da residente Befehle RAM-Speicher belegen, hängt die Anzahl der Befehle, die Sie resident machen können, von der Größe des RAM-Speichers in Ihrem System ab. Mit dem Befehl LIST können Sie jederzeit feststellen, wieviel Speicherplatz ein Befehl in etwa beanspruchen würde, den Sie resident machen möchten. Beispiel: Mit

der Eingabe **LIST C:COPY** erhalten Sie eine Ausgabe, die ungefähr wie folgt aussieht:

```
Directory "Sys:C" on Monday 15-Jun-92
copy        5496 --p-rwed 03-Jun-92 17:22:02
```

Die Größe der Datei wird rechts vom Dateinamen angezeigt. Der Befehl COPY würde als residenter Befehl also ungefähr 5,5 KB RAM beanspruchen.
Um möglichst viel Hauptspeicher für Ihre Anwendungen freizuhalten, machen Sie nur diejenigen AmigaDOS-Befehle und -Programme resident, die sie häufig benötigen und die nicht in die Workbench integriert sind. Dazu gehören ASSIGN, ED, STATUS, Format und DiskCopy. Wenn Sie genügend Hauptspeicherplatz haben und AmigaDOS regelmäßig benutzen, können Sie auch Befehle resident machen, für die es eine Entsprechung auf der Workbench gibt, z. B. COPY, DELETE, DIR, LIST, MAKEDIR und RENAME.

Machen Sie keine Befehle resident, die Sie nicht so oft benötigen, wie etwa den Startbefehl ADDBUFFERS. Die folgenden Befehle können nicht resident gemacht werden: BINDDRIVERS, CONCLIP, IPREFS, LOADRESOURCE, LOADWB und SETPATCH.

Weitere Informationen zum resident Machen von Befehlen finden Sie in der Beschreibung zum Befehl RESIDENT in Kapitel 6.

C.2 Vorab Ressourcen laden

Das vorbereitende Laden der Ressourcen in den freien Arbeitsspeicher macht diese verfügbar, ohne daß jedesmal, wenn die Ressourcen benötigt werden, die Diskette eingelegt werden muß. Mit dem Befehl LOADRESOURCE können Sie Bibliotheken, Gerätetreiber, Zeichensätze und Kataloge in den RAM laden. Diese Ressourcen bleiben so lange im RAM, wie sie benötigt werden oder bis das System sie aufgrund von Hauptspeichermangel aus dem RAM entfernt. Ressourcen, die keine Geräte sind, können im RAM auch fest verankert ("verriegelt") werden und bleiben dann auf jeden Fall so lange verfügbar, bis Sie sie wieder entriegeln.

Weitere Informationen zum Befehl LOADRESOURCE siehe in Kapitel 6.

C.3 Die Option PATH des Befehls ASSIGN

Auch mit Hilfe der Option PATH des Befehls ASSIGN können Sie sich einige Diskettenwechsel ersparen. Gewöhnlich sucht AmigaDOS auf der Original-Startdiskette nach allen Befehlen, Gerätetreibern, Bibliotheken und sonstiger benötigter Systemsoftware. Wenn nun aber eine andere Diskette eingelegt ist, erscheint ein Dialogfenster, das Sie auffordert, die Original-Startdiskette einzulegen, und zwar selbst dann, wenn sich die vom System benötigte Datei auch auf der zur Zeit eingelegten Diskette befindet.

Zur Verwendung der Option PATH sollten Sie die folgenden Befehle in Ihre Skript-Datei "User-startup" aufnehmen:

```
ASSIGN DEVS: DF0:Devs PATH
ASSIGN C: DF0:C PATH
ASSIGN L: DF0:L PATH
ASSIGN FONTS: DF0:Fonts PATH
```

Danach kopieren Sie diese Verzeichnisse von der Workbench-Diskette auf die Anwendungsdisketten, für die sie benötigt werden.

Weitere Informationen zur Option PATH des Befehls ASSIGN finden Sie in der Beschreibung des Befehls ASSIGN in Kapitel 6.

C.4 Dateien von der Workbench-Diskette löschen

Das Kopieren von Druckertreibern, Schriftarten oder Voreinstellungseditoren auf die Workbench-Diskette macht diese zwar sofort verfügbar; allerdings kann leicht das Dialogfenster **`Volume is full`** (Datenträger ist voll) erscheinen, wenn Sie versuchen, zu viel auf Ihre Workbench-Diskette zu kopieren. Es ist aber möglich, Dateien von der Workbench-Diskette zu löschen, um Platz für andere Dateien zu schaffen.

Hinweis Wenn Sie mit der Workbench-Diskette arbeiten, verwenden Sie nicht die mit dem System zusammen gelieferten Originaldisketten, sondern stets eine Sicherungskopie davon. Die Workbench-Originaldiskette sollte niemals verändert werden, damit Sie mit dieser ggf. eine Datei wiederherstellen oder die gesamte Workbench erneut installieren können.

Beachten Sie bitte, daß Sie mit dem Löschen von Systemsoftware die Möglichkeiten Ihres Amiga einschränken. Dies kann auch zu Fehlern führen, wenn eine Anwendung eine zuvor gelöschte Datei benötigt. Wenn Sie ein unerwartetes Dialogfenster oder einen Fehler bemerken, versuchen Sie, die Operation mit der Workbench-Originaldiskette zu wiederholen. Wenn dann das Problem nicht mehr auftritt, wissen Sie, daß die gelöschte(n) Datei(en) die Ursache des Fehlers war und wiederhergestellt werden muß/müssen.

Notieren Sie sich stets alle Änderungen, die Sie an Arbeitsdisketten vornehmen. Es empfiehlt sich auch, eine ECHO-Ausgabe in die Datei "User-startup" der betreffenden Diskette zu schreiben, die darauf hinweist, daß Sie nicht mit der standardmäßigen Workbench-Diskette arbeiten.

C.4.1 Dateien, die Sie löschen können

- Wenn Sie Dateien von der Workbench-Diskette löschen, beginnen Sie stets mit den weniger wichtigen Dateien, z. B. dem Programm Clock, und mit Programmen, für die Ihr System keine Verwendung hat, z. B. C:MAGTAPE, wenn Sie ohne Bandlaufwerk arbeiten, und System/NoFastMem, wenn Sie keinen Fast-Speicher besitzen.
- Durch das Löschen des AmigaDOS-Texteditors C:EDIT schaffen Sie ca. 18 KB Platz auf der Diskette.
- Löschen Sie L:CrossDOSFileSystem und DEVS:mfm.device, wenn Sie CrossDOS nicht benötigen; das bringt ca. 32 KB Speicherplatz.
- Der Inhalt des Verzeichnisses Classes macht ca. 115 KB aus; gehen Sie aber beim Löschen vorsichtig vor, da das Verzeichnis

Unterverzeichnisse enthält, auf die andere Programme zugreifen. Das Verzeichnis Classes/DataTypes wird z. B. von MultiView verwendet und sollte nicht gelöscht werden, wenn Sie die Absicht haben, mit diesem Programm Dateien anzuzeigen. Das Verzeichnis Classes/Gadgets wird vom Voreinstellungseditor "Palette" benutzt. Sie dürfen es daher nicht löschen, wenn Sie Ihre Farbeinstellungen verändern möchten.

C.4.2 Dateien, die nicht gelöscht werden dürfen

Die folgenden Dateien bzw. Verzeichnisse sollten Sie unter keinen Umständen löschen:

- Sämtliche Verzeichnisse auf der Workbench-Diskette; löschen Sie nur Dateien innerhalb von Verzeichnissen, keine kompletten Verzeichnisse
- DEVS:parallel.device
- DEVS:printer.device
- DEVS:serial.device
- LIBS:asl.library
- LIBS:commodities.library
- LIBS:diskfont.library
- LIBS:iffparse.library
- LIBS:locale.library (wenn Sie mit einer anderen Sprache bzw. länderspezifischen Datei als united_states arbeiten)
- LIBS:68040.library (Wenn Sie eine Commodore 68040-Platine haben)
- S:Startup-sequence
- L:Port-handler
- Alle nicht internen Befehle, die in der standardmäßigen Datei Startup-sequence vorkommen

Löschen Sie nur, was aus Platzgründen unbedingt sein muß. Falls Sie über den Zweck einer Datei nicht genau Bescheid wissen, löschen Sie sie nicht!

Wenn Sie unbedingt noch weiteren Speicherplatz benötigen, könnten Sie die Dateien löschen, die zur ARexx-Programmiersprache gehören:

alles in REXXC, System/RexxMast, System/RexxMast.info, LIBS:rexxsyslib.library und LIBS:rexxsupport.library. Damit gewinnen Sie ca. weitere 50 KB. Es wird allerdings davon abgeraten, diese Dateien zu löschen, da viele Anwendungsprogramme ARexx verwenden und auf die genannten Dateien zugreifen können müssen.

C.5 Die RAM-Disk

RAM: oder die RAM-Disk ist ein Bereich des internen Speichers Ihres Amiga, der als Dateispeichergerät (wie z. B. eine Disk) konfiguriert ist. Er wird auf dem Workbench-Bildschirm durch das RAM-Disk-Piktogramm dargestellt. Dateien, Verzeichnisse oder (ausreichender Speicherplatz vorausgesetzt) ganze Disketten können nach RAM: kopiert und dort vorübergehend gespeichert werden.

Die Größe von RAM: variiert dynamisch und ist immer gerade so groß wie sein Inhalt, d. h. sie ist immer zu 100 % voll. Die maximale Größe hängt jeweils vom freien allgemeinen Speicherplatz ab, ist also begrenzt.

Der wichtigste Vorteil von RAM: liegt in der hohen Zugriffsgeschwindigkeit. Da der Zugriff elektronisch und nicht mechanisch erfolgt, werden Daten in kürzester Zeit gespeichert und aufgerufen. Andererseits gehen alle in RAM: (temporär) gespeicherten Daten verloren, sobald der Computer ausgeschaltet bzw. neu gestartet wird.

Anwendungen bedienen sich des RAM: in erster Linie zum Speichern temporärer Dateien, die während des Programmablaufs erstellt werden, und zum Speichern von Sicherungskopien, die angelegt werden, wenn das Programm verlassen wird. RAM: kann auch zum Speichern von Versuchs-Skriptdateien genutzt werden, wenn Sie Befehle testen wollen oder wenn die Erstellung einer Datei auf Diskette zu langsam, riskant oder unpraktisch wäre.

C.5.1 Kopieren von einer Diskette auf eine andere

Die effizienteste Möglichkeit, bei einem System mit einem Diskettenlaufwerk Daten von einer Diskette auf eine andere zu kopieren, liegt in der Nutzung der RAM-Disk:

1. Kopieren Sie die Daten von der Quelldiskette auf die RAM-Disk.
2. Nehmen Sie die Quelldiskette aus dem Laufwerk.
3. Legen Sie die Zieldiskette ein.
4. Kopieren Sie die Daten aus der RAM-Disk auf die Zieldiskette.

Beim Speichern wichtiger Dateien in RAM: ist Vorsicht geboten. Bei einer Unterbrechung der Stromversorgung, einem Softwarefehler oder einem erneuten Starten des Computers geht alles verloren, was zu diesem Zeitpunkt in RAM: gespeichert war. Sie sollten also bei der Arbeit mit RAM: regelmäßig Sicherungskopien wichtiger Dateien auf Diskette anlegen.

Hinweis: Anders als bei anderen Kopiervorgängen können Sie eine Diskette nicht nach RAM: kopieren, indem Sie einfach das Piktogramm der Quelldiskette mit der Maus auf das RAM-Disk-Piktogramm ziehen. Statt dessen müssen Sie zunächst das RAM-Disk-Piktogramm öffnen, um anschließend das Disketten-Piktogramm in das RAM-Disk-Fenster zu ziehen. Dadurch wird eine Schublade mit dem Namen und dem Inhalt der kopierten Diskette erstellt.

C.5.2 Die reset-feste RAM-Disk

AmigaDOS verfügt auch über eine "reset-feste" (engl. recoverable) RAM-Disk, die normalerweise unter dem Namen RAD: angemeldet ist. Der Inhalt von RAD: übersteht normalerweise auch einen Neustart und auch die meisten Softwarefehler, so daß Arbeitsdateien dort in der Regel sicherer aufgehoben sind als in RAM:. (Das Ausschalten des Computers führt allerdings auch zum Verlust der Daten in RAD:.)

Im Gegensatz zu RAM: wird RAD: nicht automatisch angelegt. Die reset-feste RAM-Disk aktivieren Sie, indem Sie auf dem RAD-Piktogramm in der Schublade Storage/DOSDrivers doppelt klicken oder, falls Sie RAD: stets zur Verfügung haben möchten, indem Sie das RAD-Piktogramm in die Schublade Devs/DOSDrivers auf der Workbench-Disk ziehen. Sobald RAD: aktiviert ist, erscheint ein Disk-Piktogramm mit dem Titel RAM_0 in der Workbench-Anzeige.

Die Größe von RAD: ist, anders als die von RAM: festgelegt. Die Größe wird im Parameter HighCyl der Mount-Datei von RAD: definiert. Sie können diese Größe aber auch verändern. Dazu geben Sie für den Parameter HighCyl einen anderen Wert ein. Der Wert 79 für HighCyl stattet RAD: mit der gleichen Kapazität aus wie eine normale 880-KB-Diskette.

C.5.2.1 Startfähige RAD:-Disk

Auf einem Amiga mit mehr als 2 MB RAM:, können Sie einen RAD:-Speicher von der Größe (d. h. Kapazität) einer Diskette anlegen - die in "MountList" konfigurierte Standardgröße. Dann könnten Sie z. B. alle Workbench-Dateien nach RAD: kopieren und alle Verzeichnisse dort zuweisen, die normalerweise der Workbench-Disk zugewiesen sind. Damit erhalten Sie eine reset-feste Workbench im RAM. So könnten Sie das System von RAD: aus neu starten, ohne auf die Workbench-Diskette zugreifen zu müssen.
Es ist auch möglich, mehrere RAD:-Geräte unterschiedlicher Größe zu konfigurieren. Kopieren Sie dazu die Mount-Datei von RAD: und ändern Sie den Namen und die Einheitennummer.

Anhang D

Komplexere AmigaDOS-Funktionen

Die Informationen in diesem Anhang richten sich an erfahrene AmigaDOS-Benutzer. In diesem Anhang werden folgende Themen behandelt:

- Anpassen des Shell-Fensters
- Anpassen der Shell-Umgebung
- Verwenden von Escape-Sequenzen
- Anpassen von Startup-Dateien
- Verwenden von PIPE:

D.1 Anpassen des Shell-Fensters

Die Shell unterstützt im Fenster "Information" des Shell-Piktogramms das Merkmal WINDOW (Tool Type), durch das Sie Größe, Position und andere Eigenschaften der Shell-Fenster individuell anpassen können. Das Merkmal WINDOW hat folgendes Format:

```
WINDOW=CON:x/y/Breite/Höhe/Titel/Option/Optionen
```

Eine vollständige Beschreibung der Shell-Fenster-Spezifikation können Sie der Beschreibung des Befehls NEWSHELL in Kapitel 6 und den Beispielen in Kapitel 8 entnehmen.

D.1.1 Public-Schirme - Option PUBSCREEN

Durch Anwendungsprogramme, die eigene Bildschirme anlegen, können Bildschirme als "Public" (öffentlich) gekennzeichnet werden. Dadurch können die Fenster anderer Anwendungen und Hilfsprogramme auf demselben Bildschirm geöffnet werden. Bei AmigaDOS-Befehlen, deren Befehlsschablone das Argument PUBSCREEN/K enthält, können die zugehörigen Fenster auf solchen Public-Schirmen geöffnet werden.

Die Schablone für den Eingabe-Voreinsteller (INPUT) enthält z. B. das Argument PUBSCREEN/K. Soll das zugehörige Fenster auf einem Public-Schirm geöffnet werden, geben Sie folgenden Befehl ein:

```
1> INPUT PUBSCREEN "Public-Schirmname"
```

Sie müssen den Namen des Public-Schirms angeben, auf dem das Fenster geöffnet werden soll. Dieser Name stimmt nicht unbedingt mit dem Namen in der Titelleiste des Bildschirms überein. Oft wird als Bildschirmname der Name der Anwendung verwendet. Lesen Sie in der Dokumentation zu der jeweiligen Anwendung nach, ob die Anwendung Public-Schirme öffnen kann und wie der zugehörige Name lautet.

D.2 Anpassen der Shell-Umgebung

Sie können die Shell-Umgebung Ihren Erfordernissen entsprechend anpassen, indem Sie die Datei "Shell-startup" ändern. Dabei handelt es sich um eine Befehlsdatei, die bei jedem Öffnen der Shell ausgeführt wird. Sie können die Datei "Shell-startup" edieren, um Alias-Namen für Befehle zu definieren oder um die Shell-Eingabeaufforderung zu ändern.

D.2.1 Verwenden von Alias-Namen

Ein Alias-Name ist eine Kurzform eines langen und/oder häufig benutzten Befehls. Er kann lokal oder global gültig sein. Lokale Alias-Namen werden in einem Shell-Fenster eingegeben und können nur hier interpretiert werden. Globale Alias-Namen werden dagegen in

der Datei "Shell-startup" definiert und können von allen Shells interpretiert werden.

Der Befehl ALIAS hat folgendes Format:

```
ALIAS <Name> <Zeichenkette>
```

Dabei steht <Name> für den Alias-Namen, der hinter der Shell-Eingabeaufforderung eingegeben werden muß, damit der Befehl ausgeführt wird. <Zeichenkette> steht für die auszuführende Befehlszeile.

Kapitel 6 können Sie eine vollständige Beschreibung des Befehls ALIAS und Kapitel 8 eine Liste nützlicher Alias-Namen entnehmen.

D.2.2 Ändern der Eingabeaufforderung

Mit dem Befehl PROMPT können Sie die Shell-Eingabeauffor-derung individuell anpassen. Die vorgebene Eingabeaufforderung besteht aus der Prozeßnummer, einem Punkt, dem aktuellen Verzeichnis, einer schließenden spitzen Klammer (>) und einem Leerzeichen:

```
1.Workbench:>
```

Sie können in die Eingabeaufforderung fast jede beliebige Information aufnehmen. Die Eingabeaufforderung kann dabei mit oder ohne Prozeßnummer und Verzeichnisinformation angezeigt werden. Außerdem kann sie den Rückgabecode des zuletzt ausgeführten Befehls enthalten. Darüber hinaus können sogar Escape-Sequenzen eingefügt werden, mit denen es u. a. möglich ist, Farbe und Art der Textdarstellung in der Eingabeaufforderung zu ändern oder die Bildschirmdaten zu löschen.

Die Anpassung der Shell-Eingabeaufforderung bietet Ihnen folgende Möglichkeiten:

- Hervorheben der Eingabeaufforderung gegenüber Befehlen und deren Ausgabedaten
- Anpassen an einen Eingabeaufforderungstyp, mit dem Sie bereits vertraut sind
- Hinzufügen weiterer Informationen
- Verkürzen der Eingabeaufforderung

Kapitel 8 enthält Beispiele zur Verwendung von Escape-Sequenzen in Eingabeaufforderungen, die deren Lesbarkeit verbessern.

Sequenz (Forts.)	Funktion (Forts.)
Esc[49m	Text vor Hintergrund in der Vorgabefarbe (Farbe 0).
Esc[#u	Maximale Zeilenlänge im Fenster: #
Esc[#t	Höchstzahl Zeilen im Fenster: #
Esc[#x	Text beginnt # Pixel vom linken Fensterrand
Esc[#y	Text beginnt # Pixel vom oberen Fensterrand

Die Escape-Sequenz wird ausgeführt, wenn Sie die Eingabetaste drücken oder die Zeichenkette mit der betreffenden Sequenz ausgegeben wird.

Bestimmte Zeichen können nicht in ihrer normalen Form in Zeichenketten für AmigaDOS-Befehle eingegeben werden. Wenn Sie diesen Zeichen ein Sternchen voranstellen, können sie dennoch in den Zeichenketten für die meisten Befehle verwendet werden:

*E	Steht innerhalb einer Zeichenkette für die Taste "Esc".
*N	Steht für den Übergang auf eine neue Zeile in der Ausgabe.
*"	Ermöglicht Anführungszeichen innerhalb einer in Anführungszeichen gesetzten Zeichenkette.
**	Steht für ein einfaches Sternchen in einer Zeichenkette.

D.4 Anpassen der Startup-Dateien

Wenn Sie den Amiga starten, wird die Befehlsdatei "Startup-sequence" ausgeführt. Diese Datei ist im Verzeichnis S: gespeichert. Über die Datei "Startup-sequence" wird Disk-Pufferspeicher zugewiesen. Außerdem werden anhand dieser Datei Gerätezuweisungen durchgeführt und die in den Voreinsteller-Editoren gespeicherten Einstellungen gelesen. Darüber hinaus werden die Funktionen aufgerufen, mit denen der Amiga arbeitsbereit konfiguriert wird.

Da jeder Fehler in der Datei "Startup-sequence" zu einem kaum behebbaren Abbruch des normalen Startvorgangs führen kann, kann nur ausdrücklich davor gewarnt werden, diese Datei zu ändern. Soll der Startvorgang geändert werden, empfiehlt es sich, eine eigene

Startup-Datei namens "User-startup" (Benutzerstart) im Verzeichnis S: zu erstellen. Wenn Sie eine Datei "User-startup" erstellen, können Sie den Startvorgang ohne das Risiko anpassen, daß der normale Startvorgang abgebrochen wird oder nicht korrekt arbeitet. Diese Datei wird automatisch aus der Startsequenz heraus aufgerufen, bevor die Workbench geöffnet wird.

Hinweis Die ursprüngliche Datei "Startup-sequence" darf nicht geändert werden. Eine Änderung dieser Datei kann zu einem kaum behebbaren Abbruch des Systemstarts führen.

Die Datei "User-startup" und die anderen Startup-Dateien im Verzeichnis S: können so geändert werden, daß bestimmte Programme beim Systemstart aufgerufen, spezifische Anfangsmeldungen angezeigt oder Shell-Fenster auf dem Workbench-Bildschirm geöffnet werden. In einer Startup-Befehlsdatei sind alle AmigaDOS-Befehle einschließlich Befehlen zum Aufrufen anderer Befehlsdateien zulässig.

Lesen Sie zunächst die kompletten Befehlsbeschreibungen in Kapitel 6 und 7, bevor Sie Änderungen an einer vorhandenen Befehlsdatei vornehmen.

D.4.1 Edieren der Startup-Dateien

Wenn Sie mit einem Diskettensystem arbeiten, verwenden Sie eine Kopie der Workbench-Diskette (nicht das Original), um Änderungen an den Startup-Dateien vorzunehmen. Wenn Ihnen beim Ändern der Startup-Dateien ein Fehler unterläuft, wird die Ausführung der Datei "Startup-sequence" abgebrochen und nur eine Shell-Eingabeaufforderung angezeigt. Im Normalfall kann allerdings durch den Befehl FAILAT 21 die Datei auch bei einem Fehler bis zum Ende ausgeführt werden.

Solange Sie nicht die vorgegebene Datei "Startup-sequence" ändern, sind schwerwiegende Fehler beim Systemstart unwahrscheinlich. Wenn Sie versuchen, das System von einer Disk zu starten, auf der sich keine Datei namens "S:Startup-sequence" befindet, oder beim Startvorgang ein schwerwiegender Fehler auftritt, wird ein Shell-Fenster angezeigt, das mit dem in Abb. D-2 vergleichbar ist.

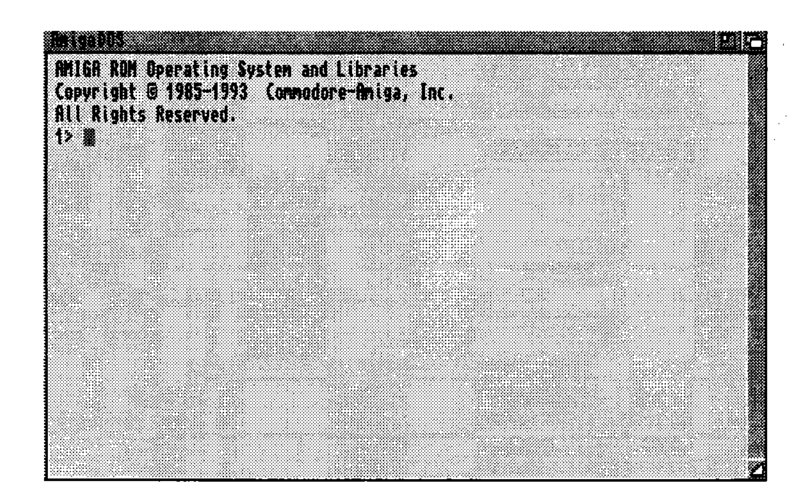


Abb. D-2. AmigaDOS-Shell-Fenster bei Startfehler

In diesem Shell-Fenster können Sie zwar Befehle eingeben, aber die üblichen Alias-Namen und der Suchpfad sind möglicherweise nicht verfügbar. In dieser Shell haben Sie die Möglichkeit, die Startup-Datei zu edieren, die zum Fehler geführt hat. Da das System jedoch nicht vollständig konfiguriert wurde, können dabei Schwierigkeiten auftreten. In der Regel ist es einfacher, das System von einer anderen Disk aus erneut zu starten.

Beim Edieren der Datei "User-startup" ist folgendes zu beachten:

- Achten Sie auf die richtige Befehlssyntax. Prüfen Sie die Wirkungsweise eines Befehls zuerst in einem Shell-Fenster, bevor Sie ihn in die Datei aufnehmen. Wenn ein Befehl in der Shell fehlerfrei ausgeführt wird, führt er wahrscheinlich auch in der Datei nicht zu Fehlern.
- Achten Sie auf die Reihenfolge der Befehle in der Befehlsdatei. Einige Befehle wie z. B. ECHO und RESIDENT können praktisch an jeder beliebigen Stelle verwendet werden. Fügen Sie jedoch keine Befehle ein, die auf Verzeichnisse oder Dateien verweisen, die bisher nicht angelegt oder zugewiesen wurden bzw. für die kein gültiger Pfad angegeben wurde.
- Geben Sie daher vorsichtshalber den vollständigen Pfad zu Verzeichnissen an, auf die zugegriffen werden soll.

- Versehen Sie Ihre Befehlsdateien mit ausreichenden Kommentaren. Text, den Sie in einer Zeile hinter einem Semikolon (;) eingeben, wird von AmigaDOS nicht verarbeitet, erscheint aber als Kommentar in der Befehlsdatei. So erkennen Sie sofort den Zweck der Zeile. Beispiel:

  ```
  ASSIGN T: RAM:T ; Hilfsverzeichnis für Skripts
  ```

- Testen Sie Änderungen an der Datei "User-startup". Wenn Sie Änderungen ausprobieren, starten Sie den Amiga nach jeder Änderung erneut, um das Ergebnis zu überprüfen. Warten Sie jedoch mit dem Neustart, bis die Datei vollständig auf die Disk geschrieben wurde. Ansonsten können Daten verloren gehen.

D.4.2 Übliche Erweiterungen der Startup-Dateien

In der folgenden Liste sind einige der Erweiterungen aufgeführt, die normalerweise in den Startup-Dateien vorgenommen werden:

- Öffnen eines Shell-Fensters
- Hinzufügen von Verzeichnissen zum Suchpfad
- Hinzufügen logischer Gerätenamen in der Zuweisungsliste
- Hinzufügen weiterer residenter Befehle
- Starten von Commodity-Exchange-Programmen und anderen Programmen
- Hinzufügen von Cache-Puffern für Laufwerke

D.5 Verwenden von PIPE:

Vom Gerät PIPE: werden Daten zwischen Programmen übertragen. Dazu wird ein temporärer Speicher im RAM: verwendet. Geben Sie PIPE: an, wenn die Ausgabedaten eines Prozesses als Eingabedaten für einen anderen Prozeß verwendet werden sollen, während der erste Prozeß noch neue Daten nachliefert. Wenn Sie zur Datenübertragung mit Pipes arbeiten, ist es bei der Ausführung von Programmen, die mit umfangreichen Dateien arbeiten, weniger wahrscheinlich, daß der verfügbare Speicherplatz nicht ausreicht. Durch PIPE: entfällt nämlich die Notwendigkeit, eine Datei, die nur

einmal erforderlich ist, zu erstellen, zu speichern und später wieder zu löschen.

Bei Verwendung von PIPE: dürfen der Quell- und der Zielprozeß nicht übereinstimmen. Beim Schreiben der Informationen in PIPE: kann ein anderer Prozeß die Daten in FIFO-Reihenfolge einlesen (FIFO =First In First Out), d. h. die Daten, die als erstes geschrieben werden, werden auch als erstes ausgelesen. Soll die Pipe benannte werden, geben Sie hinter PIPE: den gewünschten Namen ein.

Die an PIPE: gesendeten Daten werden in einen temporären Zwischenspeicher gestellt. Eine andere Anwendung liest die Daten der Pipe in der Reihenfolge ein, in der sie eingegangen sind. Die Pipe verbleibt im RAM, bis deren gesamter Inhalt eingelesen oder das System neu gestartet wird.

Von PIPE: wird pro Pipe-Name ein Puffer mit einer Mindestkapazität von 4 KB verwendet. Wenn die maximale Pufferkapazität erreicht ist, werden von PIPE: keine weiteren Daten mehr akzeptiert, bis von einem anderen Prozeß Daten aus diesem Puffer eingelesen werden. Wenn von einem Prozeß Daten aus einem Puffer eingelesen werden, wartet er, bis weitere Daten an den Puffer gesendet werden. Die Puffergröße kann nur beim ersten Verweis auf eine bestimmte Pipe vorgegeben werden.

Bei PIPE: sind folgende Optionen möglich:

channel_name Eindeutiger Kanalname. Der Name darf nicht mit einer Zahl beginnen. "channel_name" ist eine wahlfreie Option. Sie können PIPE: ohne diese Option verwenden, wenn nicht mehrere Pipes verwendet werden sollen.

buf_size Größe des zuzuweisenden Puffers in Byte. (Die Standardgröße ist 4 KB.) "buf_size" ist eine wahlfreie Option.

max_buffers Maximal zulässige Anzahl von Puffern. Der Ausgabekanal wird vorübergehend unterbrochen, wenn diese Anzahl überschritten wird. "Max_buffers = 0" bedeutet, daß es keine Obergrenze für die Anzahl der zuzuweisenden Puffer gibt.

PIPE: kann von anderen Programmen wie z. B. Textverarbeitungsprogrammen und Terminalprogrammen verwendet werden. Sie können einen beliebigen Pipe-Namen angeben. Wenn die Datei von der Anwendung sequentiell gelesen wird, können Sie PIPE:<Name>

eingeben, und die Datei wird von der Anwendung wie jede andere Datei gelesen.

Die Daten einer Pipe können zu einer anderen Pipe kopiert werden. Beispiel:

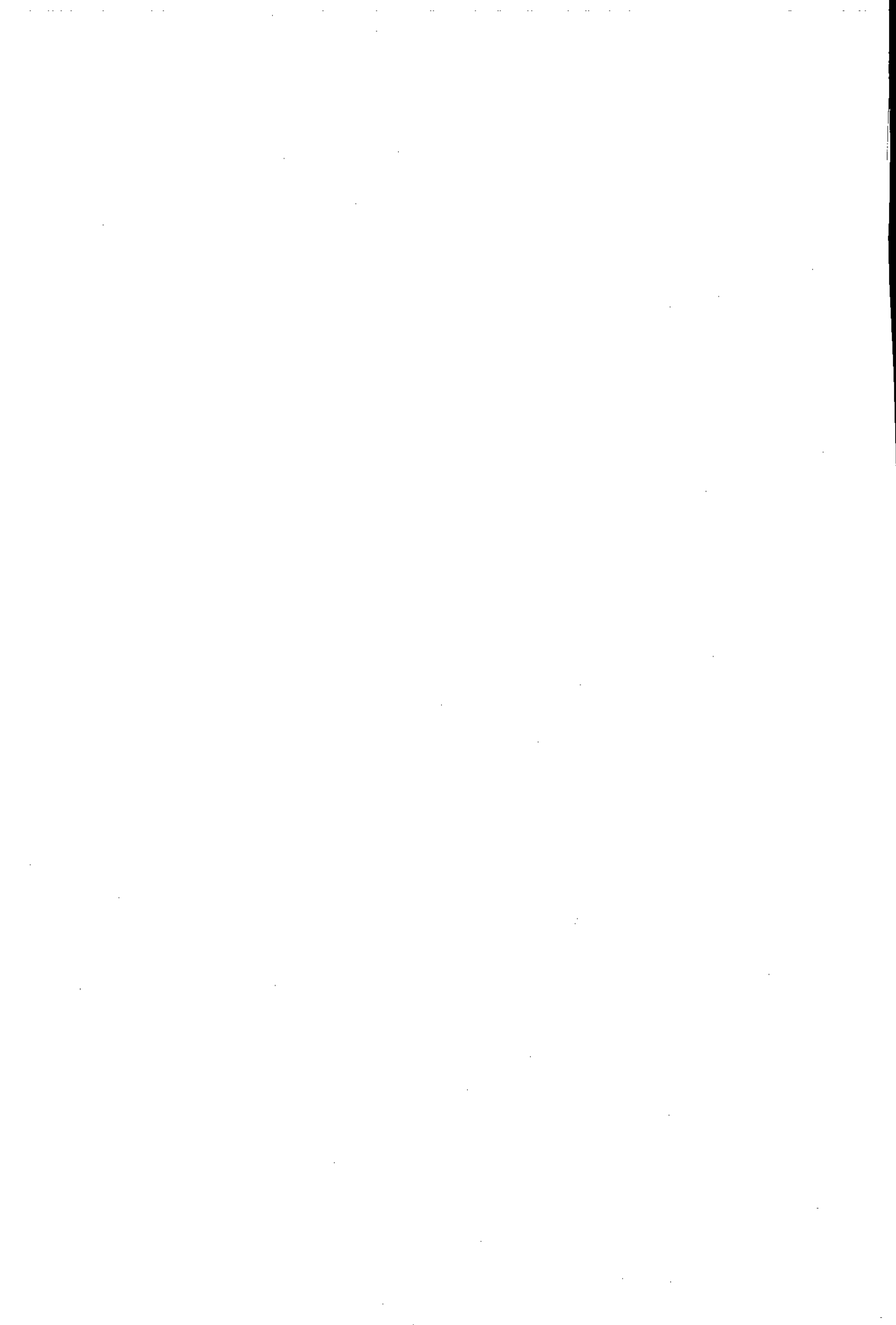
Shell-Fenster 1:	COPY Gross_Dat PIPE:a
Shell-Fenster 2:	COPY PIPE:a PIPE:b
Shell-Fenster 3:	COPY PIPE:b PIPE:c
Shell-Fenster 4:	COPY PIPE:c PIPE:d
Shell-Fenster 5:	COPY PIPE:d PIPE:e
Shell-Fenster 6:	TYPE PIPE:e ;Die Datei "Gross_Dat" wird angezeigt.

Die DOSDrivers-Anmeldedatei für PIPE: sieht wie folgt aus:

```
Handler     = L:Queue Handler
Priority    = 5
StackSize   = 3000
GlobVec     = -1
```

PIPE: wird standardmäßig während der vorgegebenen Startsequenz angemeldet.

Kapitel 8 enthält weitere Beispiele zur Verwendung von PIPE:.



Glossar

Abkoppeln (detach)

Das Trennen eines Programms von dem Prozeß, in welchem es aufgerufen wurde, so daß das Shell-Fenster des Prozesses geschlossen werden kann, bevor der Programmlauf abgeschlossen ist.

Absoluter Pfad

Ein Pfad, der vollständige Angaben zur Position einer Datei macht, einschließlich der Namen des Datenträgers bzw. der Einheit, Unterverzeichnissen und der Datei selbst.

Aktuelles Fenster

Das hervorgehobene Fenster, in das Daten über die Tastatur eingegeben werden können. Ein Shell-Fenster ist das aktuelle Fenster, nachdem es geöffnet wurde.

Aktuelles Verzeichnis

Der aktuelle Standort einer Shell innerhalb der Verzeichnisstruktur. Dieses Verzeichnis dient als Standardverzeichnis für die Ausführung von Befehlen.

Aliasname

Ein alternativer Name für einen AmigaDOS-Befehl. Aliasnamen können zum Abkürzen häufig verwendeter Befehle oder zum Ersetzen von Standardbefehlsnamen durch andere Namen verwendet werden.

Archiv

Eine Sicherungskopie einer oder mehrerer Dateien oder eines ganzen Datenträgers.

Argument
Ein Dateiname, eine Option oder andere Informationen, die zum Befehl in einer Befehlszeile angegeben werden. Auch als Parameter bezeichnet.

Bedingungskennzeichen
Eine Variable, die angibt, mit welchem Erfolg ein Befehl endet und in einer bedingten IF-Anweisung abgefragt werden kann.

Befehls-Cache
Ein Speichertyp bei 68020er-, 68030er- und 68040er-Mikroprozessoren, der eine schnellere Ausführung von Maschinenbefehlen (Instruktionen) ermöglicht.

Befehlsvorgeschichte
Eine Liste der zuletzt ausgegebenen Befehlszeilen. Diese Befehlszeilen können wieder aufgerufen, ediert und erneut ausgegeben werden.

Begrenzungszeichen
Dienen in MEmacs zum Auffinden von Anfang und Ende der Verschachtelungsstruktur eines Programms. Zulässige Begrenzungszeichen sind verschiedene Klammern (runde, eckige, spitze und geschweifte Klammern).

Begrenzungszeichen
Zeichen, die den Anfang und das Ende einer Argumentzeichenkette markieren. Im Texteditor ED sind z. B. die Zeichen ", /, \, !, :, +, - und % gültige Begrenzungszeichen.

Betriebssystem
Basissoftware, die die Funktionen eines Computers steuert.

Block

1. Eine zusammenhängende Bytegruppe (normalerweise 512 Byte) auf einem Speichermedium.
2. Ein zusammenhängender Abschnitt einer Skript-/Programmdatei. Z. B. ein IF-ENDIF-Block.

Boot-Block

Ein Bereich auf einer Disk oder PCMCIA-Karte, der den Boot-Code enthält, den das System beim Starten lesen soll. Wenn ein gültiger Boot-Block auf einer Disk bzw. Karte vorhanden ist, spricht man von einer boot- oder startfähigen Disk/Karte.

Brückenkarte

Eine von Commodore hergestellte Erweiterungskarte, durch die der Amiga PC-kompatibel wird.

Cache-Speicher

Ein Speicherbereich, der aus sehr schnellen RAM-Chips besteht. Er dient als Pufferspeicher zwischen einem schnellen CPU-Chip und einem langsameren Systemspeicher. Ist Bestandteil von Systemen mit größeren Prozesssoren.

Chip-RAM

Der Bereich des RAM, auf den der Spezial-Chip-Satz des Amiga zugreifen kann. Sämtliche akustischen und Video-Daten werden hier gespeichert.

CLI (Command Line Interface - Befehlszeilenschnittstelle)

Siehe Shell.

Cursor

Ein hervorgehobenes Rechteck am Bildschirm zur Anzeige der Schreibposition in der aktuellen Datei.

Dateisystem

Ein Teil des Betriebssystems, der definiert, wie Daten auf Speichereinheiten gespeichert werden. Dazu gehören Kopfblöcke von Dateien und Unterverzeichnissen, Datensektoren und Bitmaps, die anzeigen, welche Sektoren auf einer Disk belegt und welche frei sind.

Daten-Cache

Hardwareeinrichtung bei 68030er- und 68040er-Mikroprozessoren, die den Speicherzugriff wesentlich beschleunigt.

Direkte Befehle

Befehle im Texteditor ED, die unmittelbar nach dem Drücken der entsprechenden Tastenkombination ausgeführt werden.

Disk-Betriebssystem
Ein Teil des Betriebssystems, der für die Verwaltung von Disks und Dateien zuständig ist.

Eingabeaufforderung
Eine spezielle, individuell anpaßbare Textzeichenkette, die stets am Anfang einer Zeile des Shell-Fensters steht und anzeigt, daß das System die nächste Befehlszeile empfangen kann.

Endlosschleife
Siehe Endlosschleife.

Erweiterte Befehle
Im Texteditor ED Befehle, die aus einem oder zwei Zeichen bestehen, in Gruppen zusammengefaßt werden können und durch die Esc-Taste eingeleitet werden.

Escape-Sequenz
Eine Folge von Steuerzeichen, die normalerweise mit dem Escape-Zeichen beginnt. Escape-Sequenzen dienen u. a. zur Steuerung des Fensterformats und des Erscheinungsbilds einer Schriftart.

Fast-RAM
Teil des System-RAM, auf den Custom-Chips keinen Zugriff haben. Da nur die CPU und bestimmte Peripheriegeräte Zugriff auf das Fast-RAM haben, ist dieses wesentlich schneller als normales RAM.

Fehlergrenzwert
Der Grenzwert, ab dem ein Rückgabewert zum Abbruch einer Folge nicht interaktiver Befehle führt.

Feld
1. Ein Bereich auf dem Bildschirm, in dem ein Variablenwert angezeigt oder eingegeben wird.
2. Der Bildschirmbereich im Hintergrund des Textes unter einem Workbench-Piktogramm. Die Farbe dieses Feldes kann mit dem Font-Editor verändert werden.

Gerätetreiber
Dateien, die Funktionen verfügbar machen, die ein Gerät für den einwandfreien Betrieb innerhalb des Systems benötigt.

Globaler Vektor (GlobeVec)
Ein Mount-Parameter, der von manchen Geräten benötigt wird.

Hauptverzeichnis (Root)
Das Haupt- oder Stammverzeichnis einer Disk, dem alle Dateien und/oder Unterverzeichnisse untergeordnet sind.

Hexadezimalsystem
Zahlensystem mit der Basis 16.

Hierarchisches Dateisystem
Ein Dateisystem, in dem Verzeichnisse sowohl andere Verzeichnisse (Unterverzeichnisse) als auch Dateien enthalten können.

Hintergrundprozeß
Ein Prozeß, der kein eigenes Fenster für Ein- und Ausgabe öffnet und der nicht die übergeordnete Shell "übernimmt".

Interaktiver Auflistungsmodus
Ein Modus des Befehls DIR, bei dem nach jedem Namen in der Verzeichnisauflistung angehalten und ein Fragezeichen angezeigt wird, hinter dem Steuerbefehle eingegeben werden können.

Interne Befehle
Befehle, die in die Shell integriert sind, die also nicht von der Disk geladen werden müssen.

Interprozeßkommunikation (IPC)
Der Mechanismus, über den zwei Programme Daten miteinander austauschen können.

Jokerzeichen
Zeichen, die eine Sonderbedeutung erhalten, wenn sie in Angaben von Dateinamen verwendet werden. Diese Zeichen werden in Namensmuster verwendet.

Kommentar

1. Eine Zeile oder ein Teil einer Zeile in einer Programmdatei, die nicht ausgeführt wird, sondern nur zur Dokumentierung bzw. Erläuterung der Funktion einer Programm- oder Skript-Datei in diese geschrieben wurde.

2. Ein kurzer beschreibender Hinweis, der mit dem Befehl FILENOTE jeder beliebigen Datei hinzugefügt werden kann.

Konsolenfenster
Fenster, die von der Shell zur Ein- und Ausgabe von Text verwendet werden.

Logisches Gerät
Ein zugeordnetes Verzeichnis oder eine Softwareeinheit, auf die wie auf ein Gerät Bezug genommen wird, das/die aber auf ein Verzeichnis bzw. einen Geräte-Handler verweist.

MountList
Eine Textdatei im Verzeichnis DEVS:, die Informationen über angeschlossene oder logisch definierte Geräte enthält. Der Befehl MOUNT verwendet diese Informationen zum Aktivieren von Geräten.

Multitasking
Die Fähigkeit, mehrere Programme (oder Tasks) quasi gleichzeitig auszuführen. So können Sie auf dem Amiga z. B. mit dem Befehl RUN mehrere Programme von dem gleichen Shell-Fenster aus starten.

Namenserweiterung
Eine mit einem Punkt beginnende Zeichenkette (z. B. .info), die zur Kennzeichnung des Dateityps ans Ende von Dateinamen angefügt wird.

Namensmuster
Das Suchen von Dateien und Verzeichnissen, deren Namen einem bestimmten Muster entsprechen. In diesem Muster können auch Jokerzeichen verwendet werden.

Nicht abgekoppelte Programme
Befehle, die die Shell belegen, während sie verarbeitet werden, so daß die Shell nicht anderweitig genutzt oder geschlossen werden kann, bis der Befehl abgeschlossen ist.

Oktalsystem
Zahlensystem mit der Basis 8.

PCMCIA
Abk. für "Personal Computer Memory Card International Association", eine Organisation, die Normen für Speicherkarten erstellt.

Pfad

Informationen, die dem System den Standort einer bestimmten Datei mitteilen. Dazu können ein Datenträger- oder Laufwerksnamen sowie Namen von Unterverzeichnissen gehören. Siehe auch Suchpfad.

Pipe

Ein IPC-Mechanismus zum Weiterleiten der Ausgabe eines Befehls als Eingabe für einen anderen.

Prozeß

Ein Programm unter dem Amiga-Betriebssystem, das zum Zugriff auf Dateien mit AmigaDOS kommuniziert. Jeder Prozeß wird durch eine Nummer identifiziert, die mit dem Befehl STATUS angezeigt werden kann.

Puffer

Ein Bereich im RAM, der zur temporären Speicherung von Daten für serielle E/A bzw. Disk-E/A oder von anderen Operationen genutzt wird.

Pur

Beschreibt einen Befehl oder ein Programm, der/das resident gemacht werden kann. Bei einer puren Datei ist das Schutzbit p gesetzt.

Resident

Residente Befehle werden im Hauptspeicher gehalten, damit sie bei nochmaliger Ausführung nicht erneut geladen werden müssen. Diese Befehle müssen "pure" Dateien sein.

Rückgabewert

Eine Zahl, die nach Ausführung eines Befehls von der Shell an diesen zurückgegeben wird und anzeigt, ob der Befehl erfolgreich oder erfolglos ausgeführt wurde.

Schablone

Eine Zeichenfolge, die die Argumente und Argumenttypen eines Shell-Befehls definiert.

Schutzbits

Attribute, die anzeigen, welchen Typ eine Datei hat und welche Operationen mit ihr ausgeführt werden können. Die Schutzbits einer Datei können mit dem Befehl LIST angezeigt werden.

SCSI (Small Computer System Interface)
Ein Schnittstellenstandard für die Verbindung von Peripheriegeräten zu einem Computersystem.

Sektor
Die kleinste Speichereinheit auf einer Disk, normalerweise 512 Byte.

Shell
Textorientierte Benutzerschnittstelle zur Eingabe von Befehlen und Ausgabe an den Benutzer in einem speziellen Shell- oder Konsolenfenster. Wird auch als Befehlszeilenschnittstelle (Command Line Interface - CLI) bezeichnet.

Stapelspeicher (Stack)
Ein Speicherbereich im Computer, den sich ein Programm für temporäre Speichervorgänge "reserviert". Ein Programm kann scheitern, wenn nicht genügend Stapelspeicher zugeordnet wurde.

Startup-sequence
Eine spezielle Skript-Datei, die nach dem Starten ("Booten") des Amiga automatisch ausgeführt wird.

Suchpfad
Die Liste der Verzeichnisse, die AmigaDOS bei der Suche nach einem Befehl durchgeht, der ohne Pfadangabe eingegeben wurde. Zum Standardsuchpfad gehören das Verzeichnis C:, das aktuelle Verzeichnis sowie einige weitere Verzeichnisse, die in der standardmäßigen Startup-Sequenz angegeben sind.

Umleitung
Das Senden der Befehlsein- oder -ausgabe an eine andere Zieladresse als die Shell; z. B. in eine Datei.

User-startup
Eine spezielle Skript-Datei, die der Benutzer selbst erstellt, um den Ablauf des Systemstarts individuell anzupassen. Wenn eine solche Datei existiert, wird sie von der Startup-sequence automatisch aufgerufen.

Verbindung (Link)
Eine Datei oder ein Verzeichnis, die/das auf eine andere Datei bzw. ein anderes Verzeichnis auf einer Disk verweist. Wenn eine

Anwendung oder ein Befehl die/das ursprüngliche Datei/Verzeichnis aufruft, wird die/das verbunden Datei/Verzeichnis verwendet (wird auch als feste Verknüpfung - hard link - bezeichnet)

Verschachtelung
Mehrere Ebenen von IF-Anweisungen innerhalb von Skripts oder Programmen oder mehrere Ebenen von Unterverzeichnissen innerhalb von Verzeichnissen.

Verzeichnis-Caching
Eine Dateisystemoption zur Beschleunigung der Auflistung des Verzeichnisinhalts.

Verzeichnisverbindung
Siehe Verbindung (Link).

Verzweigungsfenster
Ein AmigaDOS-Shell-Fenster, das von MEmacs aus mit dem MEmacs-Befehl "New-Cli" geöffnet wird. AmigaDOS-Befehle, die in dieses Fenster eingegeben werden, haben keinen Einfluß auf die laufende MEmacs-Sitzung.

Weiche Verbindung (soft link)
Eine Datei oder ein Verzeichnis, die/das einen Verweis über Datenträgergrenzen hinweg darstellt. Wird bisher auf dem Amiga nicht unterstützt.

Wiederaufrufbare Befehle (reentrant)
Befehle, die die voneinander unabhängige, gleichzeitige Verwendung durch zwei oder mehr Programme unterstützen.

Wiederausführbare Befehle (re-executable)
Befehle, die nicht erneut geladen werden müssen, um mehrmals ausgeführt zu werden.

Zeichenkette, Zeichenfolge
Eine Reihe von Textzeichen, die als Einheit behandelt werden, z. B. eine vom Befehl ECHO ausgegebene Meldung. Eine Zeichenkette erfordert in der Regel Begrenzungszeichen, z. B. Leerzeichen oder Anführungszeichen, zur Kennzeichnung von Anfang und Ende der Zeichenkette.

Zeilenfenster
Unterabschnitte der Zeile, für die der Zeileneditor EDIT alle nachfolgenden Befehle ausführt.

Zeitmarke

Datum und Uhrzeit einer Datei. Diese Daten geben an, wann die Datei erstellt oder nach Änderungen das letzte Mal gespeichert wurde.

Zirkuläre Verknüpfung

Eine kreisförmig geschlossene Kette aus mehreren Verknüpfungen.

Zuweisen

Die Angabe eines Pfads zu einem Verzeichnis bzw. einer Datei unter einem zusätzlichen Namen. Damit werden logische Geräte definiert, die das Betriebssystem verwendet, z. B. C: (hier speichert AmigaDOS Befehlsdateien) oder S:, dort werden Skript-Dateien gespeichert.

Zwischenspeicher (Clipboard)

Ein Bereich im Speicher, der zum temporären Speichern von Daten beim Ausschneiden und Einfügen dient.

Index

Ä

B

C

D

E

F

N

O

Ö

P

Q

R

S

V

W

Z

Notizen